Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations	1. Thesis and Dissertation Collection, all items

1990-03

Noise adaptation and correlated maneuver gating of an extended Kalman filter

Spehn, Stephen L.

Monterey, California: Naval Postgraduate School

http://hdl.handle.net/10945/34862

This publication is a work of the U.S. Government as defined in Title 17, United States Code, Section 101. Copyright protection is not available for this work in the United States.

Downloaded from NPS Archive: Calhoun

http://www.nps.edu/library

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

DTIC FILE COPY

AD-A221 713

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DTIC
ELECTE
MAY 21 1990
S D
E

NOISE ADAPTATION AND CORRELATED MANEUVER GATING OF AN EXTENDED KALMAN FILTER

by

Stephen L. Spehn

March 1990

Thesis Co-Advisor: Hal A. Titus
Thesis Co-Advisor: Herschel H. Loomis, Jr.

Approved for public release; distribution is unlimited

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No 0704-0188

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
2b DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) EC	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS (City, State, and ZIP Code)	7b ADDRESS (City, State, and ZIP Code)
8a NAME OF FUNDING / SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification)
NOISE ADAPTATION AND CORRELATED MANEUVER GATING OF AN EXTENDED KALAMN FILTER

12 PERSONAL AUTHOR(S)
SPEHN, Stephen L.

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month, Day)	15 PAGE COUNT
Engineer's Thesis	FROM ___ TO ___	1990 March	148

16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

17 COSATI CODES		
FIELD	GROUP	SUB-GROUP

18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Kalman Filter; Maneuver Gating; Noise Adaptation

This thesis

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

Extended Kalman filtering is used to provide estimates of the position and velocity of a target based upon observations of the target's bearing and range. Non-stationary noise is shown to degrade the performance of the filter and cause filter divergence. By estimating the noise power from the variance of the filter's residual we adapt the filter to compensate for varying noise power. We also introduce the method of correlated maneuver gating to adapt the Kalman filter to target dynamics. By spatially and temporally correlating the Mahalanobis Distance of the residual, the Kalman filter's performance is increased while tracking tangentially accelerating targets. Monte Carlo simulations are run for three different sets of target dynamics: stationary, moving linearly, and accelerating tangentially. Results for the simulation show significant performance advantages of using correlated maneuver gating in conjunction with noise adaptation. These results should generalize to other applications of the extended Kalman filter whose state and observation spaces enjoy a one-to-one mapping.

(KR)

20 DISTRIBUTION / AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL TITUS, Hal A./LOOMIS, Herschel H.	22b TELEPHONE (Include Area Code) 408-646-2560/3214 22c OFFICE SYMBOL EC/Ts EC/Lm

DD Form 1473, JUN 86 Previous editions are obsolete SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603 UNCLASSIFIED

Approved for public release; distribution is unlimited.

Noise Adaptation and Correlated Maneuver Gating of an Extended Kalman Filter

by

Stephen L. Spehn
Captain, United States Marine Corps
B.S., United States Naval Academy, 1980

Submitted in partial fulfillment of the requirements for the degrees of

MASTER of SCIENCE in ELECTRICAL ENGINEERING
and
ELECTRICAL ENGINEER

from the

NAVAL POSTGRADUATE SCHOOL
March, 1990

Author: _____
Stephen L. Spehn

Approved By: _____
Hal A. Titus, Thesis Co-Advisor

_____
Herschel H. Loomis, Jr., Thesis Co-Advisor

_____
John P. Powers, Chairman,
Department of Electrical and Computer Engineering

_____
Dean of Faculty and Graduate Studies

ABSTRACT

Extended Kalman filtering is used to provide estimates of the position and velocity of a target based upon observations of the target's bearing and range. Non-stationary noise is shown to degrade the performance of the filter and cause filter divergence. By estimating the noise power from the variance of the filter's residual we adapt the filter to compensate for varying noise power. We also introduce the method of correlated maneuver gating to adapt the Kalman filter to target dynamics. By spatially and temporally correlating the Mahalanobis Distance of the residual, the Kalman filter's performance is increased while tracking tangentially accelerating targets. Monte Carlo simulations are run for three different sets of target dynamics: stationary, moving linearly, and accelerating tangentially. Results for the simulations show significant performance advantages of using correlated maneuver gating in conjunction with noise adaptation. These results should generalize to other applications of the extended Kalman filter whose state and observation spaces enjoy a one-to-one mapping.

Accession For
NTIS GRA&I
DTIC TAB
Unannounced
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1

TABLE OF CONTENTS

I. INTRODUCTION 1

A. TARGET TRACKING 1

B. OBJECTIVES OF THIS REPORT 1

C. REPORT ORGANIZATION 2

II. KALMAN FILTER 4

A. DEFINITION OF TERMS 4

B. SYSTEM MODEL 5

C. LINEAR, RECURSIVE FORM 6

D. MEASURE OF OPTIMALITY 7

E. PROJECTION OF ERROR COVARIANCE 9

F. RESIDUAL 10

G. KALMAN GAINS 11

H. KALMAN FILTER EQUATIONS 13

III. EXTENDED KALMAN FILTER 14

A. SYSTEM MODEL 14

B. FIRST-ORDER APPROXIMATION 15

C. ITERATED LINEARIZATION 21

D. COMMENTS ON OPTIMALITY 22

E. EXTENDED KALMAN FILTER EQUATIONS 23

IV. MODEL OF THE TRACKING SYSTEM 24

A. PHYSICAL SYSTEM 24

B. STATE SPACE MODEL 25

C. OBSERVATION MATRIX 27
D. NOISE 29
E. DEFINITION OF ERROR FUNCTION 31
F. SIMULATION SCENARIOS 31
V. NOISE ADAPTATION 36
A. NOISE SENSITIVITY 36
B. ESTIMATING OBSERVATION NOISE 39
C. NOISE ADAPTATION 43
VI. MANEUVER GATING 45
A. RESIDUAL GATING 45
B. ERROR COVARIANCE ADJUSTMENT 48
C. CORRELATED GATING 49
VII. SIMULATIONS 53
A. NOISE ADAPTATION 53
B. MANEUVER GATING 59
VIII. RESULTS 82
A. NOISE ADAPTATION 82
B. MANEUVER GATING 83
IX. CONCLUSIONS AND RECOMMENDATIONS 86
APPENDIX A. TRANSFORMATION OF NOISE PROCESSES 87
APPENDIX B. ERROR ELLIPSES 94
APPENDIX C. PROGRAM CODE 103
LIST OF REFERENCES 138
BIBLIOGRAPHY 139
INITIAL DISTRIBUTION LIST 140

ACKNOWLEDGEMENT

I would like to thank my advisor, Hal Titus, for having the courage and confidence to let a wild horse run free. His guidance was always supportive; his counsel was always timely; his jokes were sometimes well delivered. I look forward to our continued work together. I would also like to express my gratitude for my friends who supported me emotionally throughout this time. Special thanks go to Carla, Lorna, and Ellen; your support was constantly felt. I treasure our friendship.

Most of all I want to thank Patricia, who came into my life offering love and understanding. Her support and confidence have given me strength; and that has made all the difference.

I. INTRODUCTION

A. TARGET TRACKING

The detection, location, and tracking of targets plays a critical role in many aspects of military operations as well as civilian aviation. The ability to accurately track and predict the movement of aircraft, for example, makes modern air-traffic-control possible, and the importance of target tracking in military operations can not be overstated. This report investigates a particularly powerful method of target tracking known as Kalman filtering.

Kalman filtering has been used with tremendous success in target tracking for almost thirty years. Its strength lies in its firm mathematical foundation of statistical optimality. We investigate the behavior of an extended Kalman filter in tracking a target by means of bearing and range observations. This is the case for most civilian and military radars.

B. OBJECTIVES OF THIS REPORT

This report treats the extended Kalman filter as a starting point, not a destination. By combining statistics and heuristics, we introduce methods which can significantly improve the performance of the extended Kalman filter. These methods are sufficiently general to apply to a wide range of applications, not just radar target tracking. The methods proposed serve to allow the Kalman filter to adapt to its environment in the face of dynamic noise statistics and maneuvering targets.

We begin by showing that the Kalman filter is only optimal when it is given correct *a priori* information about the noise sources affecting the system. The performance of the filter degrades as the noise power varies from the *a priori* estimates. We propose a method to improve the performance of the Kalman filter in the presence of unpredicted or non-stationary noise processes. This method uses the residual (or innovations) provided by the Kalman filter to estimate the statistical properties of the noise. These parameters are then used adaptively in the Kalman equations.

We also show that the ability of a Kalman filter to track a tangentially accelerating target is improved greatly by maneuver gating. Although maneuver gating has been discussed before in the literature, we propose gating off of the Mahalanobis Distance of the residual. This gating statistic is then processed for spatial and temporal correlation. It is this correlated gating process which provides increased accuracy and divergence rejection to the Kalman filter.

C. REPORT ORGANIZATION

This report is organized into six major sections. The first is this introduction, which serves as a guide to approaching this report. Chapters II and III give a mathematical derivation of the extended Kalman filter equations. Chapter IV models the physical tracking system which is used as the basis for the simulations throughout this report. Chapters V and VI describe the methods which we propose for improving the performance of the extended Kalman filter. Chapters VII through IX show the simulations and give our results and conclusions. The appendices give supportive mathematical treatments and program code.

It is recommended that the chapters be taken in the order they are presented. The reader who is conversive with Kalman filtering, however, should skim Chapters II

through IV and proceed quickly to Chapter V. Appendices A and B give mathematical treatment to topics for which we were unable to find suitable derivations in the literature. These appendices should prove useful for other researchers interested in this area, but are not required to appreciate the results of this research. Finally, the program code listed in Appendix C is purely for the benefit of others who are conducting work in target tracking.

II. KALMAN FILTER

The Kalman filter has been used extensively in the design of estimation systems since it was first presented by Kalman and Bucy [Ref. 1-2] in 1960. Its rapid acceptance and subsequent success are due, in part, to the Kalman filter's recursive nature and provable optimality for certain systems. Also, the Kalman filter has the desirable quality of maintaining the physical meaning of the system dynamics by utilizing a state space representation.

The provable optimality mentioned above holds for systems which are linear and time-invariant (LTI), and corrupted by additive white Gaussian noise (AWGN). This chapter will develop the Kalman filter equations for just such a system. The development will follow closely that given by Gelb in Chapter IV of Ref. 3 and starts by defining the terms involved in the derivation. After that, we model the physical system and the noise processes involved. Then the form of the filter will be argued, rather than proved. Finally, after choosing criteria for optimality, the equations which optimize the performance of this filter will be derived.

A. DEFINITION OF TERMS

All of the terms used in this chapter are defined in Table (2.1). These terms are explained further when they are introduced in the derivation. Terms with a single time subscript (i.e., x_k) refer to the value of the term at that time. Those terms with dual subscripts (i.e., $P_{k+1|k}$) refer to the value of the term at the time of the first subscript given observations through the second. Since no special annotation is used to denote a matrix, the third column gives the dimension of each entry for its matrix form. The dimensions are given first in the table.

TABLE 2.1: DEFINITION OF TERMS

System order:	n	
Observation size:	m	
Identity matrix	I	n x n
System state:	x_k	n x 1
Transpose of x:	x^T_k	1 x n
State transition matrix:	ϕ	n x n
State excitation noise:	w_k	n x 1
Observation:	z_k	m x 1
Observation matrix:	H	m x n
Observation noise:	v_k	m x 1
State estimate:	$\hat{x}_{k+1\|k}$	n x 1
Estimate error:	$\tilde{x}_{k+1\|k}$	n x 1
Expected value of the error:	$E[\tilde{x}_{k+1\|k}]$	n x 1
Error covariance matrix:	$P_{k+1\|k+1}$	n x n
Residual:	r_{k+1}	m x 1

B. SYSTEM MODEL

In order to estimate the parameters of a system, it is necessary to have a good model of that system. The state space model of our linear system is given in Equation (2.1), and the measurements are described by Equation (2.2). This is a standard state space matrix representation for a system of linear, discrete-difference equations.

$$x_{k+1} = \phi x_k + w_k \tag{2.1}$$

$$z_{k+1} = Hx_{k+1} + v_{k+1} \tag{2.2}$$

The physical state of the system (position, velocity, etc.) is described by x_k, and the observed parameters (bearing, range, etc.) are contained in z_k.

This system is time-invariant because neither ϕ nor H is dependent on the time subscript k. The noise processes are assumed to be stationary, independent, zero-mean, AWGN. Although this is only an idealization, it is often justified in real systems. The statistical properties of the noise processes are given in Equations (2.3) through (2.7).

$$E[w_k] = 0 \tag{2.3}$$

$$E[w_j w^T_k] = Q\delta_{jk} \tag{2.4}$$

$$E[v_k] = 0 \tag{2.5}$$

$$E[v_j v^T_k] = R\delta_{jk} \tag{2.6}$$

$$E[w_j v^T_k] = 0 \tag{2.7}$$

The Kronecker delta function, δ, is defined by

$$\delta_{jk} = \begin{cases} 1 & j = k \\ 0 & j \neq k \end{cases}. \tag{2.8}$$

The matrices Q and R in Equations (2.4) and (2.6) are the covariance matrices for the noise processes. For this system, the noise covariance matrices are non-zero diagonal matrices which denote the power present in the noise. This simple, linear model will be generalized in Chapter 3.

C. LINEAR, RECURSIVE FORM

Before deriving the Kalman filter equations we will first determine the form of the filter. The form we assume is that of a two-step, linear, recursive filter, as shown in Equations (2.9) and (2.10). This form is both simple and efficient. The current estimate is a linear combination of the previous estimate and the current observation.

Since the filter does not store the observations, it requires only a fixed amount of memory to process an arbitrary number of observations.

$$\hat{x}_{k+1|k} = K_1 \hat{x}_{k|k} \tag{2.9}$$

$$\hat{x}_{k+1|k+1} = K_2 \hat{x}_{k+1|k} + K_3 z_{k+1} \tag{2.10}$$

Although this form is assumed for its simplicity, Ref. 1 demonstrates that it is optimal for a linear system.

D. MEASURE OF OPTIMALITY

An optimal system is generally considered to be any system which minimizes a cost function (or maximizes a performance function) subject to specific constraints. For our system, we desire a filter which gives unbiased estimates that minimize a function of the estimate error. Constants K_1 in Equation (2.9) and K_2 in Equation (2.10) are chosen to make the estimate unbiased. The constant K_3 is chosen to minimize a cost function related to the expected error.

An unbiased estimate is an estimate whose error has an expected value of zero. This is an entirely reasonable desire for a filter which we intend to call optimal. The estimate errors are given as

$$\tilde{x}_{k+1|k} = x_{k+1} - \hat{x}_{k+1|k} \tag{2.11}$$

and

$$\tilde{x}_{k+1|k+1} = x_{k+1} - \hat{x}_{k+1|k+1} \tag{2.12}$$

and their expected values satisfy

$$E[\tilde{x}_{k+1|k+1}] = E[\tilde{x}_{k+1|k}] = 0. \tag{2.13}$$

The value of K_1 can be determined by substituting Equations (2.1) and (2.9) into Equation (2.11) and then adding another term to both sides to yield

$$\tilde{x}_{k+1|k} - \phi\hat{x}_{k|k} = \phi x_k + w_k - K_1\hat{x}_{k|k} - \phi\hat{x}_{k|k}. \tag{2.14}$$

By regrouping terms and taking the expected value of both sides, Equation (2.14) can be written in a form which can make use of Equation (2.13).

$$E[\tilde{x}_{k+1|k}] = E[\phi\tilde{x}_{k|k}] + E[w_k] + (\phi - K_1)\hat{x}_{k|k} \tag{2.15}$$

Since the values of the expectations are zero by Equations (2.3) and (2.13), we get

$$K_1 = \phi. \tag{2.16}$$

So the new form of Equation (2.9) is

$$\hat{x}_{k+1|k} = \phi\hat{x}_{k|k}. \tag{2.17}$$

Proceeding in a similar fashion, we substitute Equations (2.1) and (2.10) into Equation (2.12) to give

$$\tilde{x}_{k+1|k+1} = \phi x_k + w_k - K_2\hat{x}_{k+1|k} - K_3 z_{k+1}. \tag{2.18}$$

Now use Equation (2.2), regroup the terms, and take the expectation, using Equation (2.13), to get the value for K_2.

$$K_2 = I - K_3 H \tag{2.19}$$

Combining Equations (2.10), (2.17), and (2.19), the form for the Kalman filter equations can be written as

$$\hat{x}_{k+1|k} = \phi\hat{x}_{k|k} \tag{2.20}$$

and

$$\hat{x}_{k+1|k+1} = (I - GH)\hat{x}_{k+1|k} + Gz_{k+1} \tag{2.21}$$

where G replaces K_s as the Kalman gain matrix. Equation (2.21) can also be written in the form

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + G(z_{k+1} - H\hat{x}_{k+1|k}). \tag{2.22}$$

Equations (2.20) and (2.22) are known as the time-update and the observation-update equations, respectively. In order to solve for the value of the Kalman gain matrix, G, we must specify the error function to be minimized. A reasonable choice would be a quadratic function of the error. So we will choose the value of G which satisfies Equation (2.23).

$$G : \min_G (J_{k+1} = E[\tilde{x}^T_{k+1|k+1}\tilde{x}_{k+1|k+1}]) \tag{2.23}$$

Solving the above equation requires an analysis of the error covariance matrices. This is handled in the next section.

E. PROJECTION OF ERROR COVARIANCE

The error covariance matrices are defined by Equations (2.24) and (2.25). These matrices give a feeling for the expected magnitude of the estimation error. Strictly, they are the covariance matrices for the errors, which are n-dimensional, zero-mean, Gaussian random vectors.

$$P_{k+1|k} = E[\tilde{x}_{k+1|k}\tilde{x}^T_{k+1|k}] \tag{2.24}$$

$$P_{k+1|k+1} = E[\tilde{x}_{k+1|k+1}\tilde{x}^T_{k+1|k+1}] \tag{2.25}$$

Rewriting Equation (2.11) as

$$\tilde{x}_{k+1|k} = \phi\tilde{x}_{k|k} + w_k \tag{2.26}$$

and putting it into Equation (2.24), we get

$$P_{k+1|k} = \phi P_{k|k}\phi^T + Q. \tag{2.27}$$

Similarly, writing Equation (2.12) as

$$\tilde{x}_{k+1|k+1} = (I - GH)\tilde{x}_{k+1|k} - Gv_{k+1} \tag{2.28}$$

and putting it into Equation (2.25), we get

$$P_{k+1|k+1} = (I - GH)P_{k+1|k}(I - GH)^T + GRG^T. \tag{2.29}$$

It is pleasing to note that the Kalman filter is able to carry along its own error analysis by means of Equations (2.28) and (2.29). All that remains is to find the value for the Kalman gain matrix, G. But to simplify the notation, we will first define the residual, and its covariance.

F. RESIDUAL

The residual is the difference between the observation and the expected value of the observation. This is given by

$$r_{k+1} = z_{k+1} - E[z_{k+1}]. \tag{2.30}$$

Since the estimate is unbiased, we see that

$$E[z_{k+1}] = E[Hx_{k+1}] + E[v_{k+1}] \tag{2.31a}$$
$$= H\hat{x}_{k+1|k}. \tag{2.31b}$$

Putting Equation (2.31b) into Equation (2.30) and expanding z_{k+1} with Equation (2.2), we get the final form for the residual,

$$r_{k+1} = Hx_{k+1} - H\hat{x}_{k+1|k} + v_{k+1} \tag{2.32a}$$

$$= H\tilde{x}_{k+1|k} + v_{k+1}. \tag{2.32b}$$

Using this definition of the residual, the observation-update equation can be written as

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + Gr_{k+1} \tag{2.33}$$

and the covariance of the residual is found to be

$$V[r_{k+1}] = E[r_{k+1}r^T_{k+1}] \tag{2.34a}$$

$$= HP_{k+1|k}H^T + R. \tag{2.34b}$$

Having this, we are ready to derive the value for the Kalman gain matrix.

G. KALMAN GAINS

Returning to Equation (2.23), we see that it can be written in terms of the error covariance as

$$G : \min_G \{J_{k+1} = \text{trace}(P_{k+1|k+1})\}. \tag{2.35}$$

This is equivalent to minimizing the length of the estimation error vector. Doing so requires that we take the partial derivative of J_{k+1} with respect to G and set it equal to zero. Solving the resulting equation will yield the optimal value of G.

Taking the partial derivative of J_{k+1} requires the use of the following matrix relation, where A and B are matrices and B is symmetric:

$$\frac{\partial}{\partial A}[\text{trace}(ABA^T)] = 2AB. \tag{2.36}$$

Putting Equation (2.29) into Equation (2.35), applying the above relation, and setting the result equal to zero gives

$$-2(I - GH)P_{k+1|k}H^T + 2GR = 0. \tag{2.37}$$

Solving for G,

$$G = P_{k+1|k}H^T(HP_{k+1|k}H^T + R)^{-1} \tag{2.38a}$$

$$= P_{k+1|k}H^TV[r_{k+1}]^{-1}. \tag{2.38b}$$

It is important to note at this point that the value of G is a function of time, and will henceforth use the time subscript k.

Before compiling a coherent set of Kalman filter equations, we will take the effort to simplify Equation (2.29). By putting Equation (2.38b) into Equation (2.29), we can get, after some manipulation,

$$P_{k+1|k+1} = P_{k+1|k} - P_{k+1|k}H^TV[r_{k+1}]^{-1}HP_{k+1|k}. \tag{2.39}$$

By expanding out the variance of the residual and using the matrix inversion lemma [Ref. 4],

$$(A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}, \tag{2.40}$$

we can, with a modicum of persistence, rewrite Equation (2.29) as

$$P_{k+1|k+1} = (I - GH)P_{k+1|k}. \tag{2.41}$$

H. KALMAN FILTER EQUATIONS

We now have a set of recursive equations which not only calculates the time-varying optimal gain matrix, but also provides a detailed error analysis of the estimate. These equations are given in Table (2.2) below.

TABLE 2.2: KALMAN FILTER EQUATIONS

Estimate projection:	$\hat{x}_{k+1\|k} = \phi\hat{x}_{k\|k}$	(2.42)
Residual:	$r_{k+1} = z_{k+1} - H\hat{x}_{k+1\|k}$	(2.43)
Error projection:	$P_{k+1\|k} = \phi P_{k\|k}\phi^T + Q$	(2.44)
Residual covariance:	$V[r_{k+1}] = HP_{k+1\|k}H^T + R$	(2.45)
Kalman gain:	$G_{k+1} = P_{k+1\|k}H^T V[r_{k+1}]^{-1}$	(2.46)
Estimate update:	$\hat{x}_{k+1\|k+1} = \hat{x}_{k+1\|k} + G_{k+1}r_{k+1}$	(2.47)
Error update:	$P_{k+1\|k+1} = (I - G_{k+1}H)P_{k+1\|k}$	(2.48)

These equations satisfy our definition of optimal estimates (unbiased estimates which minimize J_{k+1}) for a system described by Equations (2.1) through (2.7). Their recursive nature makes them ideal for software implementation, and requires only that they be given the following initial conditions;

initial estimate:	$\hat{x}_{0\|0}$,	(2.49)
and error covariance:	$P_{0\|0}$.	(2.50)

It is important to remember that the equations of Table (2.2) are only statistically optimal when the system is linear, and perfect *a priori* knowledge of the noise exists. The next chapter will consider the case for a non-linear system with perfect *a priori* noise statistics.

III. EXTENDED KALMAN FILTER

Chapter II showed that the Kalman filter yields a statistically optimal estimate for the states of a linear, time-invariant (LTI) system which is corrupted by additive, white, Gaussian noise (AWGN). Unfortunately, many real systems of interest, such as the system studied in this report, are not LTI. This chapter will consider the time-invariant system which has a linear state-transition equation, but a non-linear observation equation. When the Kalman equations are applied in this fashion to a non-linear system, they are called the extended Kalman filter equations.

To derive the extended Kalman filter, we first define the state equations for the non-linear system. Then we linearize the Kalman filter equations derived in Chapter II to produce a reasonable estimate of the state. Finally, we give one method for improving the accuracy of the linearization, and hence the estimate.

A. SYSTEM MODEL

The state space representation for a system with a non-linear observation equation is given in Equations (3.1) and (3.2). We see that the only difference between this system and the LTI system of Equations (2.1) and (2.2) is that the observation matrix, H, is now a function of the state.

$$x_{k+1} = \phi x_k + w_k \tag{3.1}$$

$$z_{k+1} = H(x_{k+1}) + v_{k+1} \tag{3.2}$$

The noise processes, w_k and v_k, are AWGN with the same statistical parameters as given in Equations (2.3) through (2.7).

Calculating the observation matrix requires an exact knowledge of the state of the system, which we do not have. But we need the value of H in order to calculate the

estimate of the state. This circular dependence requires an approach which can separate the calculations without creating excessive approximation errors. The approach we chose is to expand the value of H in a power series about the current estimate, and then maintain only the first order terms.

B. FIRST-ORDER APPROXIMATION

Power-series expansion can be applied to any function which possesses derivatives of all orders in the region of expansion. The expansion of the function f(x) about the point r is given as

$$\hat{f}_N(x) = \sum_{n=0}^{N} \frac{1}{n!} f^n(x)|_{(x=r)}(x - r)^n \tag{3.3}$$

where $f^n(x)|_{(x=r)}$ is the n^{th} derivative of f with respect to x, evaluated at x = r. This expansion is valid because the error between the summation and the actual value tends to zero as N tends to infinity.

$$\lim_{N\to\infty} [\hat{f}_N(x) - f(x)] = 0. \tag{3.4}$$

Performing this expansion on the observation matrix about the time-update for the estimate and keeping only the first two terms gives

$$\hat{H}(x_{k+1}) = H(\hat{x}_{k+1|k}) + \frac{\partial}{\partial x^T}_{k+1} H(\hat{x}_{k+1|k})[x_{k+1} - \hat{x}_{k+1|k}] \tag{3.5a}$$

$$= H(\hat{x}_{k+1|k}) + \frac{\partial}{\partial x^T}_{k+1} H(\hat{x}_{k+1|k})\ \tilde{x}_{k+1|k}. \tag{3.5b}$$

With Equation (3.5b) we are now able to proceed with the Kalman filter analysis in a fashion similar to Chapter II.

1. Linearization

Our choice of point about which to linearize H is critical to the derivation of the Kalman equations. The point we chose, the time-update of the estimate (also called the estimate projection), is the estimate based on everything up to, but not including, the current observation. This is the best choice for the linearization because it accounts for all available information just prior to needing H for further calculations.

Methods for improving the accuracy of this linearization include keeping second-order terms in the series expansion and iterating over the point of expansion. The former, while mathematically sound, can prove difficult to implement due to the need to calculate the second partial derivative of a matrix. For this reason, we chose to implement an iterative approach which is discussed later in this chapter.

2. Observation Matrix

Before we begin substituting the linearized observation matrix into the linear Kalman equations, we will employ one more term for notational simplicity. By defining

$$H' \equiv \frac{\partial}{\partial x^T}\Big|_{k+1|k} H(\hat{x}_{k+1|k}) \tag{3.6}$$

we can write Equation (3.5b) as

$$\hat{H}(x_{k+1}) = H(\hat{x}_{k+1|k}) + H'\tilde{x}_{k+1|k} \tag{3.7}$$

where the time dependency for H' is understood to be k+1|k. This notation will allow us to write the extended Kalman filter equations in a form similar to the ones derived in Chapter II. An interesting (and important) consequence of Equation (3.7) can be found by taking the expectation of both sides of it, as seen in Equation (3.8). This relation will prove critical to the development of the extended Kalman filter equations.

$$E[H(x_{k+1})] = H(\hat{x}_{k+1|k}) \tag{3.8}$$

3. Linear, Recursive Form

For simplicity and efficiency we choose a linear, recursive set of equations for our estimates, just as we did in Chapter II. In this case, however, we start with only the estimate-update equation,

$$\hat{x}_{k+1|k+1} = a_{k+1|k} + G_{k+1}z_{k+1} \tag{3.9}$$

and then solve for $a_{k+1|k}$ by requiring that the estimate errors be unbiased. Defining the estimate errors to be

$$\tilde{x}_{k+1|k} = x_{k+1} - \hat{x}_{k+1|k} \tag{3.10}$$

and

$$\tilde{x}_{k+1|k+1} = x_{k+1} - \hat{x}_{k+1|k+1} \tag{3.11}$$

and substituting Equations (3.2), (3.9), and (3.10) into Equation (3.11) yields

$$\tilde{x}_{k+1|k+1} = \tilde{x}_{k+1|k} + \hat{x}_{k+1|k} - a_{k+1|k} - G_{k+1}(H(x_{k+1}) + v_{k+1}). \tag{3.12}$$

Taking the expectation of both sides of Equation (3.12) gives the value for $a_{k+1|k}$ as

$$a_{k+1|k} = \hat{x}_{k+1|k} - G_{k+1}H(\hat{x}_{k+1|k}). \tag{3.13}$$

By defining the residual as the difference between the observation and the expected value of the observation,

$$r_{k+1} = z_{k+1} - E[z_{k+1}]$$

$$= z_{k+1} - H(\hat{x}_{k+1|k}) \tag{3.14}$$

we can combine Equations (3.9), (3.13), and (3.14) to yield the standard extended Kalman filter estimate-update equation,

$$\hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + G_{k+1}r_{k+1}. \tag{3.15}$$

4. Projection of Error Covariance

Having the form of the extended Kalman equations, we can now proceed in a manner similar to that of Chapter II. Using Equations (3.10) and (3.11), and the definitions of error covariance

$$P_{k+1|k} = E[\tilde{x}_{k+1|k}\tilde{x}^T_{k+1|k}] \tag{3.16}$$

and

$$P_{k+1|k+1} = E[\tilde{x}_{k+1|k+1}\tilde{x}^T_{k+1|k+1}], \tag{3.17}$$

we will derive the error-projection and error-update equations. Putting Equation (3.1) into Equation (3.10) and using

$$\hat{x}_{k+1|k} = \phi\hat{x}_{k|k} \tag{3.18}$$

we can write Equation (3.10) as

$$\tilde{x}_{k+1|k} = \phi\tilde{x}_{k|k} + w_k. \tag{3.19}$$

Now using Equation (3.19) in (3.16) we get

$$P_{k+1|k} = \phi P_{k|k}\phi^T + Q. \tag{3.20}$$

Finding the equation for $P_{k+1|k+1}$ will require the use of the vector gradient of H, as given in Equation (3.6). Starting with Equation (3.11) and substituting into it Equations (3.9) and (3.13) gives

$$\tilde{x}_{k+1|k+1} = x_{k+1} - \hat{x}_{k+1|k} - G_{k+1}[z_{k+1} - H(\hat{x}_{k+1})]. \tag{3.21}$$

By using Equations (3.2) and (3.7), we can put the error into the form

$$\begin{aligned}\tilde{x}_{k+1|k+1} &= x_{k+1} - \hat{x}_{k+1|k} \\ &\quad - G_{k+1}[H(x_{k+1}) + v_k] \\ &\quad + G_{k+1}H(\hat{x}_{k+1|k}) \\ &= x_{k+1} - \hat{x}_{k+1|k} \\ &\quad - G_{k+1}v_k \\ &\quad - G_{k+1}[H(x_{k+1}) - H(\hat{x}_{k+1|k})] \\ &= \tilde{x}_{k+1|k} - G_{k+1}v_k \\ &\quad - G_{k+1}H'\tilde{x}_{k+1|k} \\ &= [I - G_{k+1}H']\tilde{x}_{k+1|k} - G_{k+1}v_k. \end{aligned} \tag{3.22}$$

This gives us an equation for $P_{k+1|k+1}$

$$P_{k+1|k+1} = (I - GH')P_{k+1|k}(I - GH')^T + GRG^T \tag{3.23}$$

which is identical in form to Equation (2.29), the covariance update for the linear Kalman filter, with the substitution of H' for H.

5. Variance of the Residual

The one final step required prior to deriving the Kalman gains is to get an expression for the variance of the residual. much as we did in Chapter II. By

substituting Equations (3.2) and (3.7) into the definition of the residual, Equation (3.14), we get

$$\begin{aligned} r_{k+1} &= H(x_{k+1}) + v_k - H(\hat{x}_{k+1|k}) \\ &= H'\tilde{x}_{k+1|k} + v_k. \end{aligned} \tag{3.24}$$

The covariance of Equation (3.24) is

$$V[r_{k+1}] = E[r_{k+1}r^T_{k+1}] \tag{3.25a}$$

$$= H'P_{k+1|k}H'^T + R. \tag{3.25b}$$

6. Kalman Gains

The similarity between the linear and non-linear error covariance equations is both pleasing and fortunate. Since Equations (3.20) and (3.23) are similar in form to Equations (2.27) and (2.29), we can use our results from Chapter II to derive the Kalman gains for the non-linear system. Specifically, the value of G_{k+1} which satisfies the following condition

$$G : \min_G \{J_{k+1} = \text{trace}(P_{k+1|k+1})\}, \tag{3.26}$$

is found by taking the partial derivative of J_{k+1} with respect to G_{k+1} and setting it equal to zero. Solving the resulting equation yields the optimal value of G_{k+1},

$$\begin{aligned} G_{k+1} &= P_{k+1|k}H'^T(H'P_{k+1|k}H'^T + R)^{-1} \\ &= P_{k+1|k}H'^TV[r_{k+1}]^{-1}. \end{aligned} \tag{3.27}$$

C. ITERATED LINEARIZATION

As mentioned earlier in this chapter, one method for improving the accuracy of the extended Kalman filter is to linearize the observation matrix iteratively about successively improved estimates of the current state. This iterative technique will be detailed in this section.

We will denote the i^{th} estimate of x_{k+1} prior to the observation at time k+1 to be

$$\hat{x}_{i,k+1|k} \tag{3.28}$$

where we initialize the iteration by setting

$$\hat{x}_{0,k+1|k} = \hat{x}_{k+1|k}. \tag{3.29}$$

We can now rewrite Equations (3.6) and (3.7) as

$$H_i' \equiv \frac{\partial}{\partial x^T}_{k+1|k} H(\hat{x}_{i,k+1|k}) \tag{3.30}$$

and

$$\hat{H}_i(x_{k+1}) = H(\hat{x}_{i,k+1|k}) + H_i'\tilde{x}_{i,k+1|k} \tag{3.31}$$

where the errors are defined as

$$\tilde{x}_{i,k+1|k} = x_{k+1} - \hat{x}_{i,k+1|k} \tag{3.32}$$

and

$$\tilde{x}_{i,k+1|k+1} = x_{k+1} - \hat{x}_{i,k+1|k+1}. \tag{3.33}$$

By combining Equations (3.13) and (3.30), and recalling that

$$E[x_{k+1}] = E[\hat{x}_{k+1|k}], \tag{3.34}$$

we get the equation for the iterated estimates,

$$\hat{x}_{i+1|k+1|k} = \hat{x}_{i|k+1|k+1}$$
$$= \hat{x}_{k+1|k} + G_{i|k+1}[z_{k+1} - H(\hat{x}_{i|k+1|k}) - H_i'(\hat{x}_{k+1|k} - \hat{x}_{i|k+1|k})]. \quad (3.35)$$

Expressing this iterative procedure as an algorithm gives insight into its simplicity and power. The algorithm is given in pseudo-code in Table (3.1).

TABLE 3.1: ITERATION ALGORITHM

calculate $P_{k+1|k}$
set $\hat{x}_{0|k+1|k} = \hat{x}_{k+1|k}$
set i = 0
do
    calculate H_i'
    calculate $G_{i|k+1|k}$
    calculate $\hat{x}_{i|k+1|k+1}$
    set $\hat{x}_{i+1|k+1|k} = \hat{x}_{i|k+1|k+1}$
    set i = i + 1
until (i > i_{max} or
    $(\hat{x}_{i+1|k+1|k} - \hat{x}_{i|k+1|k}) < \epsilon$)
calculate $P_{k+1|k+1}$

The criteria for stopping the iteration will usually be when either i (the iteration index) has reached a preset limit or the difference between successive iterated estimates is below a specified tolerance. With this algorithm, and the equations previously derived, we will present the extended Kalman filter equations after a brief observation about optimality.

D. COMMENTS ON OPTIMALITY

We derived the extended Kalman filter equations by using the linear Kalman equations of Chapter II as a guide. While we showed that the linear equations were statistically optimal, we make no similar claims for the non-linear case. We cannot, in fact, claim that the equations in Table (3.2) will converge to a reasonable solution. What

we do state is that the approach used to derive these equations was based on the same principles and procedures as was used for the linear Kalman equations. Based upon this and previous experience, it is reasonable to assume that the equations given in Table (3.2) will yield a good, though sub-optimal, estimate of the state.

E. EXTENDED KALMAN FILTER EQUATIONS

The complete set of iterated, first-order, extended Kalman filter equations are given in Table (3.2). The iteration index, i, goes from zero to some preset limit. This limit is set only to limit computational expense.

TABLE 3.2: ITERATED, FIRST-ORDER, EXTENDED KALMAN EQUATIONS

Error projection:	$P_{k+1\|k} = \phi P_{k\|k}\phi^T + Q$	(3.36)
Estimate projection:	$\hat{x}_{k+1\|k} = \phi\hat{x}_{k\|k}$	(3.37)
Iteration seed:	$\hat{x}_{0,k+1,k} = \hat{x}_{k+1\|k}$	(3.38)
Begin Iteration		
Observation matrix:	$H_i' = \frac{\partial}{\partial x^T}_{k+1\|k} H(\hat{x}_{i,k+1\|k})$	(3.39)
Residual:	$r_{i,k+1} = z_{k+1} - H(\hat{x}_{i,k+1\|k})$	(3.40)
Residual covariance:	$V[r_{i,k+1}] = H_i' P_{k+1\|k} H_i'^T + R$	(3.41)
Kalman gain:	$G_{i,k+1} = P_{k+1\|k} H_i' V[r_{i,k+1}]^{-1}$	(3.42)
Estimate update:	$\hat{x}_{i,k+1\|k+1} = \hat{x}_{i,k+1\|k} + G_{i,k+1} r_{k+1} - H_i'(\hat{x}_{k+1\|k} - \hat{x}_{i,k+1\|k})$	(3.43)
End Iteration		
Error update:	$P_{k+1\|k+1} = (I - G_{k+1}H')P_{k+1\|k}$	(3.44)

IV. MODEL OF THE TRACKING SYSTEM

In this chapter we introduce the mathematical model of the tracking system which we will use for the Monte Carlo simulations throughout the remainder of this report. We start by presenting the physical relationship between the target and the observer. We then show how this can be represented in the state space for use with the extended Kalman filter equations derived in Chapter IV. Following that, the noise processes involved are explained and the appropriate equations derived. Having a state space description of the noisy tracking system, we then define the error function used as a test statistic for performance of the Kalman filter. Finally, we explain the five scenarios which were used for the Monte Carlo simulations.

A. PHYSICAL SYSTEM

The system used in this report consists of a single observer and a single target. The coordinate system is a two-dimensional Cartesian coordinate system with the observer at the origin and the target initially located in the first (upper right) quadrant as shown in Figure (4.1). The x and y axes correspond to East and true North, respectively. The target is free to move unrestricted throughout the coordinate space, while the observer is stationary and remains at the origin.

The observation information received by the observer consists of bearing and range measurements from the observer to the target. This is consistent with the measurements supplied by most types of radar in use today. These measurements are taken at equal time intervals of 10 seconds (T = 0.002778 hours). This simple system will now be transformed into state space notation.

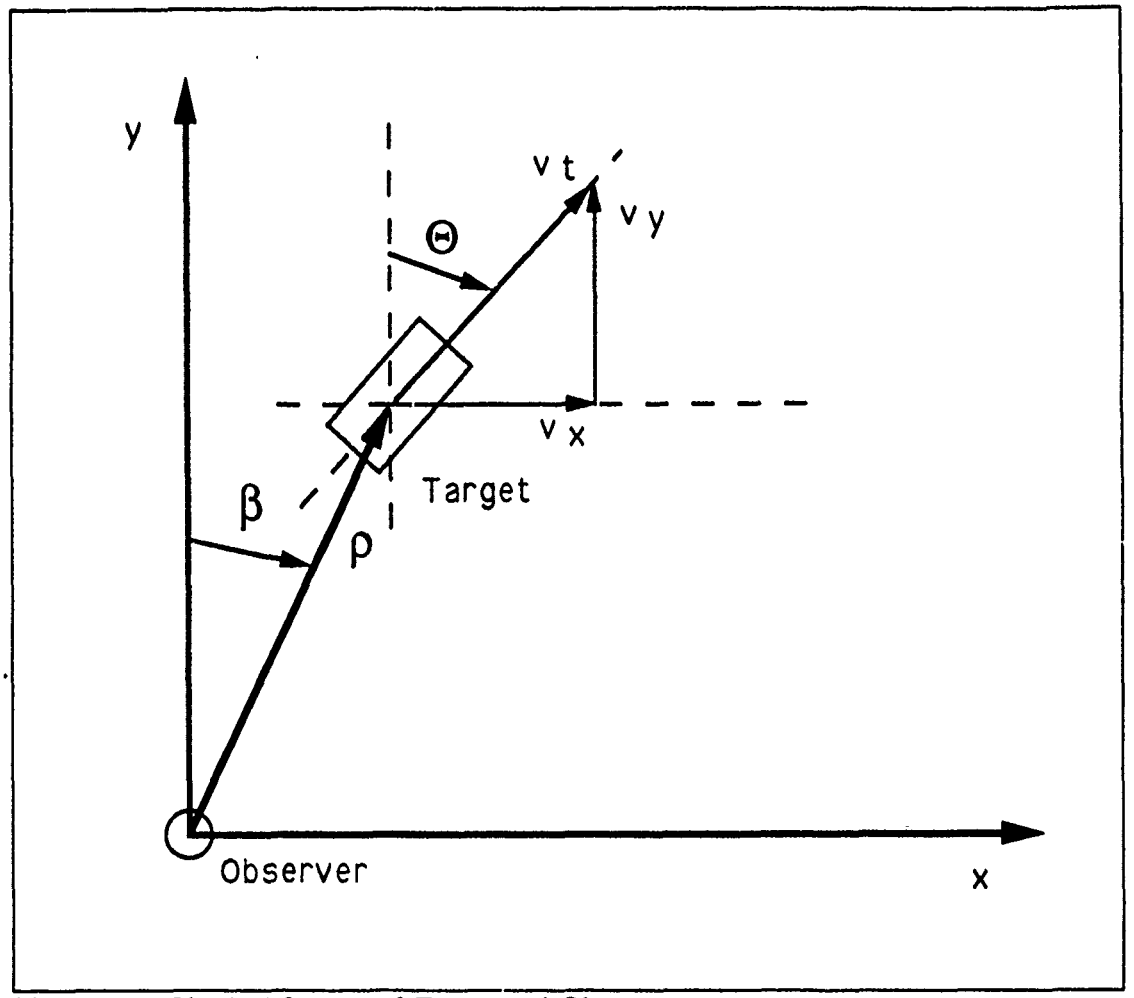

Figure 4.1: Physical Layout of Target and Observer

B. STATE SPACE MODEL

The discrete-time, state space model of the system depicted by Figure (4.1) is

$$\underline{X}_{k+1} = \phi \underline{X}_k + w_k \qquad (4.1)$$

where the state vector, $\underline{X}$, is defined to contain the minimum number of elements necessary to describe the target in terms of its degrees of freedom. As such, $\underline{X}$ consists of the position and velocity of the target. (The capital letter, $\underline{X}$, is used in this chapter

to denote the state to preclude confusion with the coordinate, x. The rest of this report conforms with the standard practice of using a lower case x for the state, as coordinates are not referenced outside of this chapter and the appendices.) The unknown (i.e., unpredictable) accelerations of the target are accounted for by the state excitation vector, w_k.

$$\underline{X} \equiv \begin{bmatrix} x \\ v_x \\ y \\ v_y \end{bmatrix} \tag{4.2}$$

The matrix ϕ in Equation (4.1) is chosen to fit the target dynamics. The two types of target dynamics which are normally assumed are stationary, and moving linearly at constant velocity. The appropriate ϕ's for these two cases are given in Equations (4.3) and (4.4).

Stationary:

$$\phi_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.3}$$

Moving Linearly:

$$\phi_2 = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

The constant, T, is the observation interval mentioned previously. Since we design for the most general case, we use ϕ_2 for all of the simulations.

Putting Equations (4.2) and (4.4) into Equation (4.1) gives the final state equation for the simulated system. We will now present the observation matrix and derive the first-order linearization required by Chapter III.

C. OBSERVATION MATRIX

The observation equation for our system consists of noise added to bearing and range measurements. Since the bearing and range to the target are non-linear functions of x and y, the observation equation is given as

$$z_{k+1} = H(\underline{X}_{k+1}) + v_{k+1} \tag{4.5}$$

where

$$z \equiv \begin{bmatrix} \beta \\ \rho \end{bmatrix}. \tag{4.6}$$

The bearing and range are defined by

bearing:
$$\beta = \tan^{-1}\left(\frac{x}{y}\right) \tag{4.7}$$

and range:
$$\rho = \sqrt{x^2 + y^2}\,. \tag{4.8}$$

The extended Kalman filter equations derived in Chapter III require the vector gradient of the observation matrix in the state space. This matrix is given as

$$H' = \frac{\partial}{\partial \underline{X}^T} H(\hat{\underline{X}}) \tag{4.9}$$

$$= \begin{bmatrix} \frac{\partial}{\partial x}\beta & \frac{\partial}{\partial v_x}\beta & \frac{\partial}{\partial y}\beta & \frac{\partial}{\partial v_y}\beta \\ \frac{\partial}{\partial x}\rho & \frac{\partial}{\partial v_x}\rho & \frac{\partial}{\partial y}\rho & \frac{\partial}{\partial v_y}\rho \end{bmatrix}_{\underline{X} = \hat{\underline{X}}} \tag{4.10}$$

where the partial derivatives are evaluated at the estimate.

By putting Equations (4.7) and (4.8) into Equation (4.10) and doing the partial differentiations we get the expressions for the elements of the H' matrix.

$$\frac{\partial}{\partial v_x}\beta = 0 \tag{4.11a}$$

$$\frac{\partial}{\partial v_y}\beta = 0 \tag{4.11b}$$

$$\frac{\partial}{\partial v_x}\rho = 0 \tag{4.11c}$$

$$\frac{\partial}{\partial v_y}\rho = 0 \tag{4.11d}$$

$$\frac{\partial}{\partial x}\beta = \left(\frac{1}{1 + \frac{x^2}{y^2}} \right) \cdot \frac{1}{y}$$

$$= \frac{y}{x^2 + y^2} = \frac{y}{\rho^2} \tag{4.12a}$$

$$\frac{\partial}{\partial y}\beta = \left(\frac{1}{1 + \frac{x^2}{y^2}} \right) \cdot \left(- \frac{x}{y^2} \right)$$

$$= - \frac{x}{x^2 + y^2} = - \frac{x}{\rho^2} \tag{4.12b}$$

$$\frac{\partial}{\partial x}\rho = \frac{1}{2} \frac{1}{\rho} 2x$$

$$= \frac{x}{\rho} \tag{4.12c}$$

$$\frac{\partial}{\partial y}\rho = \frac{1}{2} \frac{1}{\rho} 2y$$

$$= \frac{y}{\rho} \tag{4.12d}$$

Using Equations (4.11) and (4.12) in Equation (4.10) gives

$$H' = \begin{bmatrix} \frac{y}{\rho^2} & 0 & \frac{-x}{\rho^2} & 0 \\ \frac{x}{\rho} & 0 & \frac{y}{\rho} & 0 \end{bmatrix}_{\underline{X} = \hat{\underline{X}}} \tag{4.13}$$

which is required for the extended Kalman filter analysis. We will now discuss the noise processes inherent in the system.

D. NOISE

The noise processes which are denoted by w_k and v_k in Equations (4.1) and (4.5) are assumed to be additive, white, Gaussian noise (AWGN) with the auto- and cross-correlation properties described by Equations (2.3) through (2.7). The covariance matrix for the observation noise is

$$R = \begin{bmatrix} \sigma_\beta^2 & 0 \\ 0 & \sigma_\rho^2 \end{bmatrix} \tag{4.14}$$

where the values used for the variations are

$$\sigma_\beta^2 = 0.0005 \text{ rad}^2 \tag{4.15}$$

and

$$\sigma_\rho^2 = 0.0005 \text{ km}^2. \tag{4.16}$$

We felt that these values were representative of actual radar accuracies and provided nominal values about which to perform the sensitivity analysis detailed in Chapter V.

The analysis of the state excitation covariance matrix, Q, is considerably more complex and is derived in Appendix A. The purpose of Q is to account for the unknown accelerations of the target. The results of the derivation are that Q is given as

$$Q = \Gamma \begin{bmatrix} E[a_x^2] & E[a_{xy}] \\ E[a_{xy}] & E[a_y^2] \end{bmatrix} \Gamma^T \tag{4.17}$$

where the matrix elements are

$$E[a_x^2] = \left(\frac{v_x}{v_t}\right)^2 \sigma_v^2 + v_y^2 \sigma_\theta^2 \tag{4.18}$$

$$E[a_y^2] = \left(\frac{v_y}{v_t}\right)^2 \sigma_v^2 + v_x^2 \sigma_\theta^2 \tag{4.19}$$

and $$E[a_x a_y] = E[a_y a_x] =$$

$$v_x v_y \left[\left(\frac{\sigma_v}{v_t}\right)^2 - \sigma_\theta^2 \right] \tag{4.20}$$

and the state noise coefficient matrix is

$$\Gamma = \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix} \tag{4.21}$$

where T = 0.002778 hours.

The purpose of Equations (4.17) through (4.20) is to take the values of the variances of linear and tangential acceleration and transform them into the Cartesian coordinate system of the state space. The values for these variances were assumed to be

(linear) $$\sigma_v^2 = 0.001 \ (\text{km/hr}^2)^2 \tag{4.22}$$

(tangential) $$\sigma_\theta^2 = 0.001 \ (\text{rad/hr}^2)^2 \tag{4.23}$$

which are characteristic of a non-maneuvering target. These values were intentionally chosen small so as to make the Kalman filter believe that the target was non-maneuvering. The effects of this choice will be shown in Chapter VII. Having a model for the noise processes in the system, we will now define a performance measure for the estimates.

E. DEFINITION OF ERROR FUNCTION

The choice of error (or performance) function is an arbitrary one. Its only purpose is to provide us with some statistic to use as a measure of filter performance. This statistic is then the value which determines the relative quality of the various filter estimates. For this report we chose a weighted sum of the filter errors as the error function.

$$\sum_{k=1}^{K_{max}} kT|x_k - \hat{x}_{k|k}| \tag{4.24}$$

By taking a weighted sum of the estimate errors we can weight the final estimates more heavily than the early estimates. The weighting factor, kT, is equal to the absolute time since the initial observation. We feel that, although this is not the only possible error function, it is certainly a reasonable one. With this error function we can now measure the performance of the Kalman filter in the various scenarios described in the next section.

F. SIMULATION SCENARIOS

For this report we chose to run the adaptive Kalman filter on five different scenarios. Each scenario is described by the positions, velocities and accelerations of the target and observer. The observer was chosen to be stationary at the origin of the Cartesian coordinate system as shown previously in Figure (4.1). The target dynamics are given in Table (4.1) and Figures (4.2) through (4.5) for each scenario. The tangential accelerations are given in g units where a g is defined as

$$g = 9.8 \text{ m/sec}^2 \tag{4.25}$$

and the observation interval, T, is 0.002778 hours (10 seconds).

TABLE 4.1: TARGET DYNAMICS BY SCENARIO

Scenario #	x_0 (km)	y_0 (km)	v_{x0} (km/hr)	v_{y0} (km/hr)	$acc_{tangential}$ (g's)	duration (sec)
1	5	37	0	0	0	0
2	5	37	450	0	0	0
3	5	37	400	-280	0	0
4	5	37	600	50	0.8	40
5	5	37	450	-300	-0.8	50

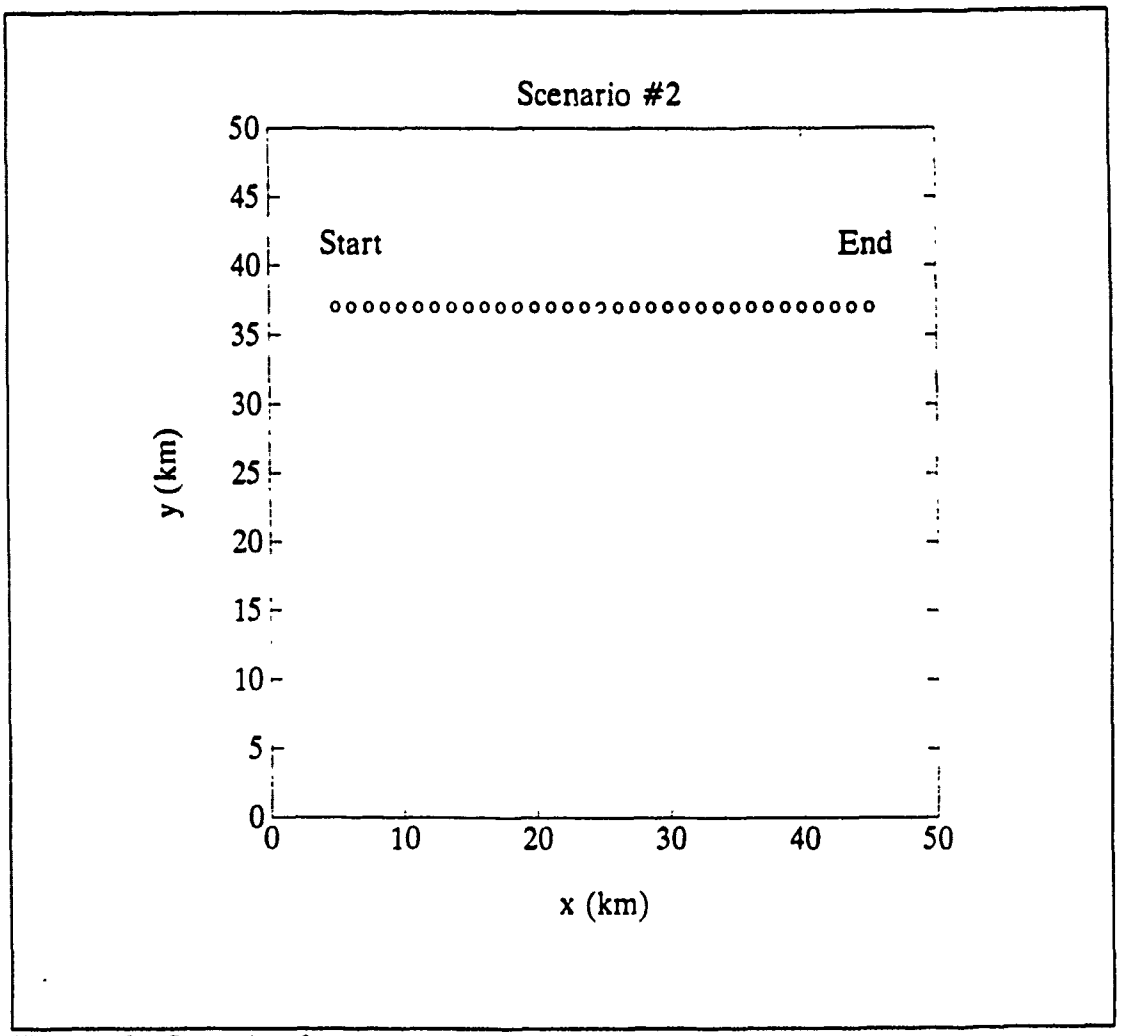

Figure 4.2: Scenario #2

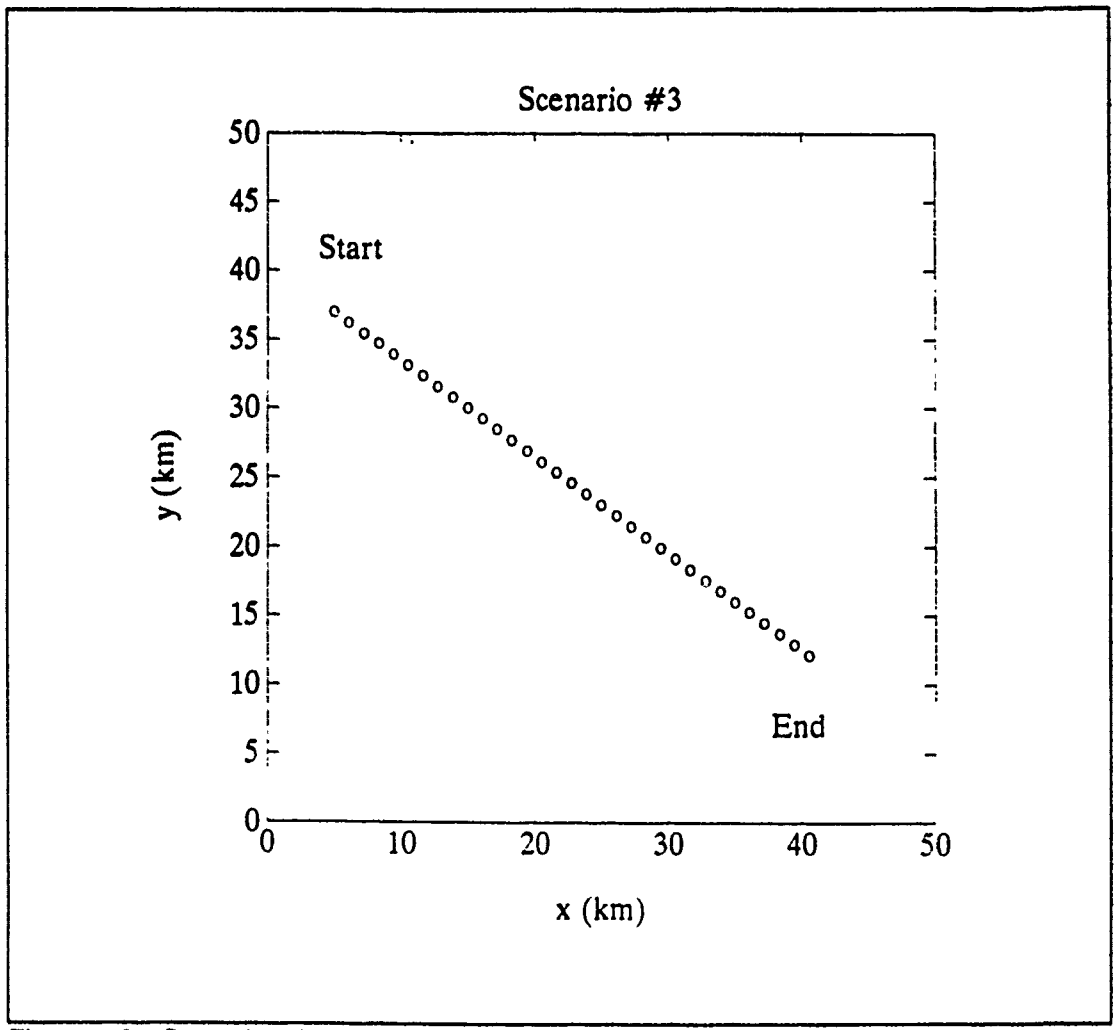

Figure 4.3: Scenario #3

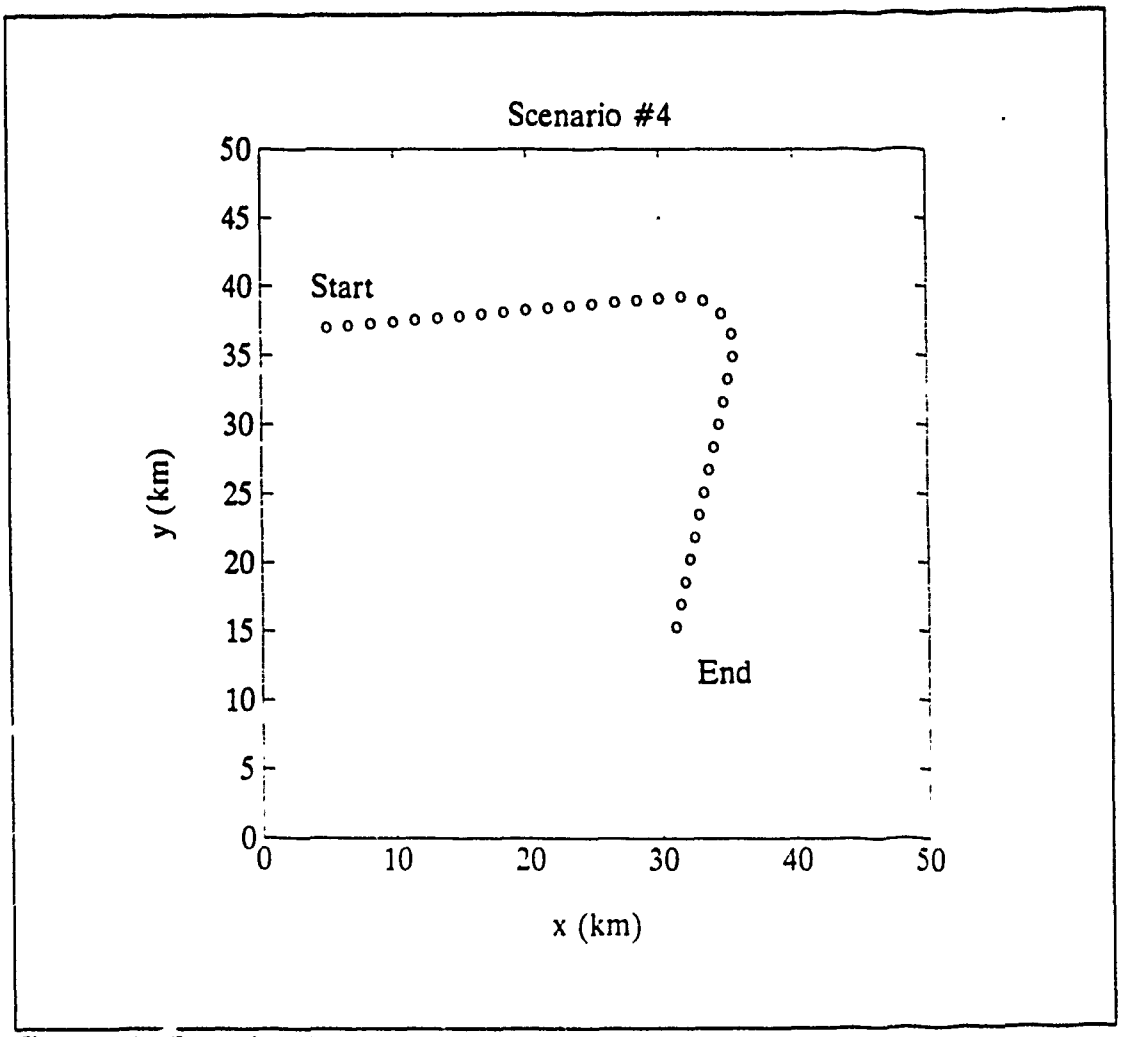

Figure 4.4: Scenario #4

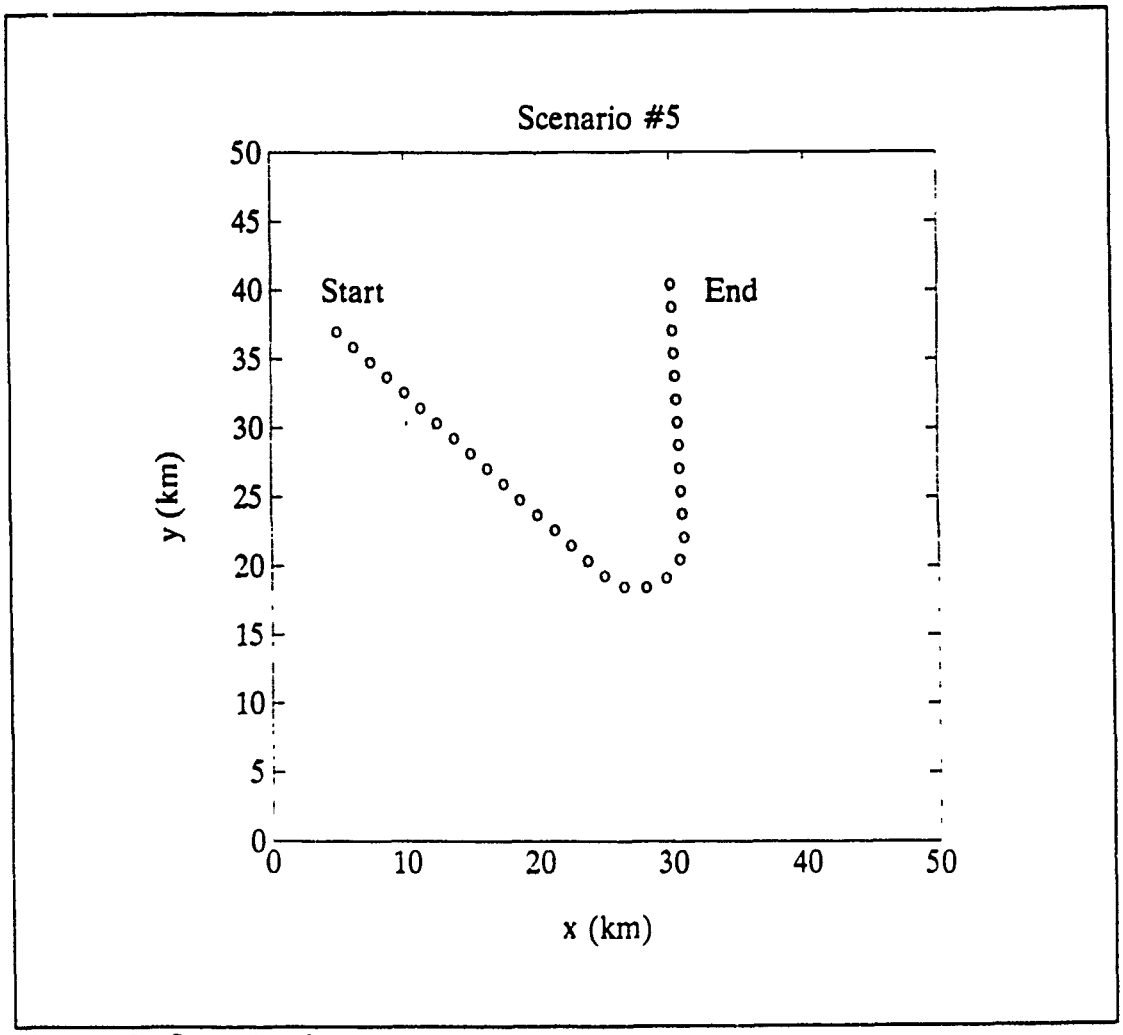

Figure 4.5: Scenario #5

We are now ready to delve into the details of the adaptive methods which we have proposed. The noise adaptation is covered in Chapter V and the adaptive maneuver gating is treated in Chapter VI. Following that, the simulations are presented in Chapter VII and the results in Chapter VIII.

V. NOISE ADAPTATION

This chapter begins by explaining the critical dependence of the Kalman filter's performance on the quality of the *a priori* noise estimates. We will both argue and demonstrate that the Kalman filter achieves its optimal performance when the *a priori* measurement noise v_k is equal to the actual measurement noise. Having shown this, we then present two methods for estimating the actual observation noise of the system. Finally, we use this estimate of the actual observation noise to adapt the Kalman filter to improve its performance.

A. NOISE SENSITIVITY

A rigorous derivation of the effects of incorrect *a priori* noise estimates on the performance of the Kalman filter is well beyond the scope of this report. We have not, in fact, been able to find such a derivation anywhere in the literature. As such, we will present our case based upon reasonability arguments and support our statements with sample simulations.

In order to proceed with this discussion, it is necessary to make a clear distinction between the Kalman parameters which are calculated with incorrect (assumed) *a priori* information and those which are calculated with correct *a priori* information. For our purposes we shall use the following notation.

assumed: $$P_{k+1|k+1} = (I - G_{k+1}H)P_{k+1|k} \tag{5.1}$$

correct: $$P^{\circ}_{k+1|k+1} = (I - G^{\circ}_{k+1}H)P^{\circ}_{k+1|k} \tag{5.2}$$

Our goal then, is not to minimize some function of Equation (5.1), but to minimize some function of the difference between Equations (5.1) and (5.2). We will use the simplest

of minimization functions, and just minimize the difference for any arbitrary value of $P_{k+1|k}$ by setting

$$P_{k+1|k} - P^{\circ}_{k+1|k} = 0. \tag{5.3}$$

Putting Equations (5.1) and (5.2) into Equation (5.3) we find that

$$\begin{aligned} P_{k+1|k+1} - P^{\circ}_{k+1|k+1} &= (G^{\circ}_{k+1} - G_{k+1}) H P_{k+1|k} \\ &= 0. \end{aligned} \tag{5.4}$$

The only way to solve Equation (5.4) for arbitrary $P_{k+1|k}$ is by setting

$$G_{k+1} = G^{\circ}_{k+1}. \tag{5.5}$$

Equation (5.5) gives the expected (and pleasing) result that the optimal estimates are achieved only if the Kalman gains are equal to the gains calculated from correct *a priori* information. Though this result is not unexpected, it is often not appreciated in physical implementations of Kalman filters.

The effect of incorrect *a priori* noise information is shown graphically in Figure (5.1). This graph was produced by setting the *a priori* noise covariance equal to the actual noise covariance scaled by the factor K_R (noise covariance scale) and running the simulations described by scenario three of Table (4.1).

$$R_{a\ priori} = K_R\ R_{actual} \tag{5.6}$$

Although the exact shape of Figure (5.1) is dependent upon the nature of the system being simulated, it is characteristic of the results we expect. Figure (5.1) shows clearly that the best performance (lowest total error) was achieved when the *a priori* R was

equal to the actual value of R (K_R = 1). This result can best be explained by consideration of the information carried by the observations.

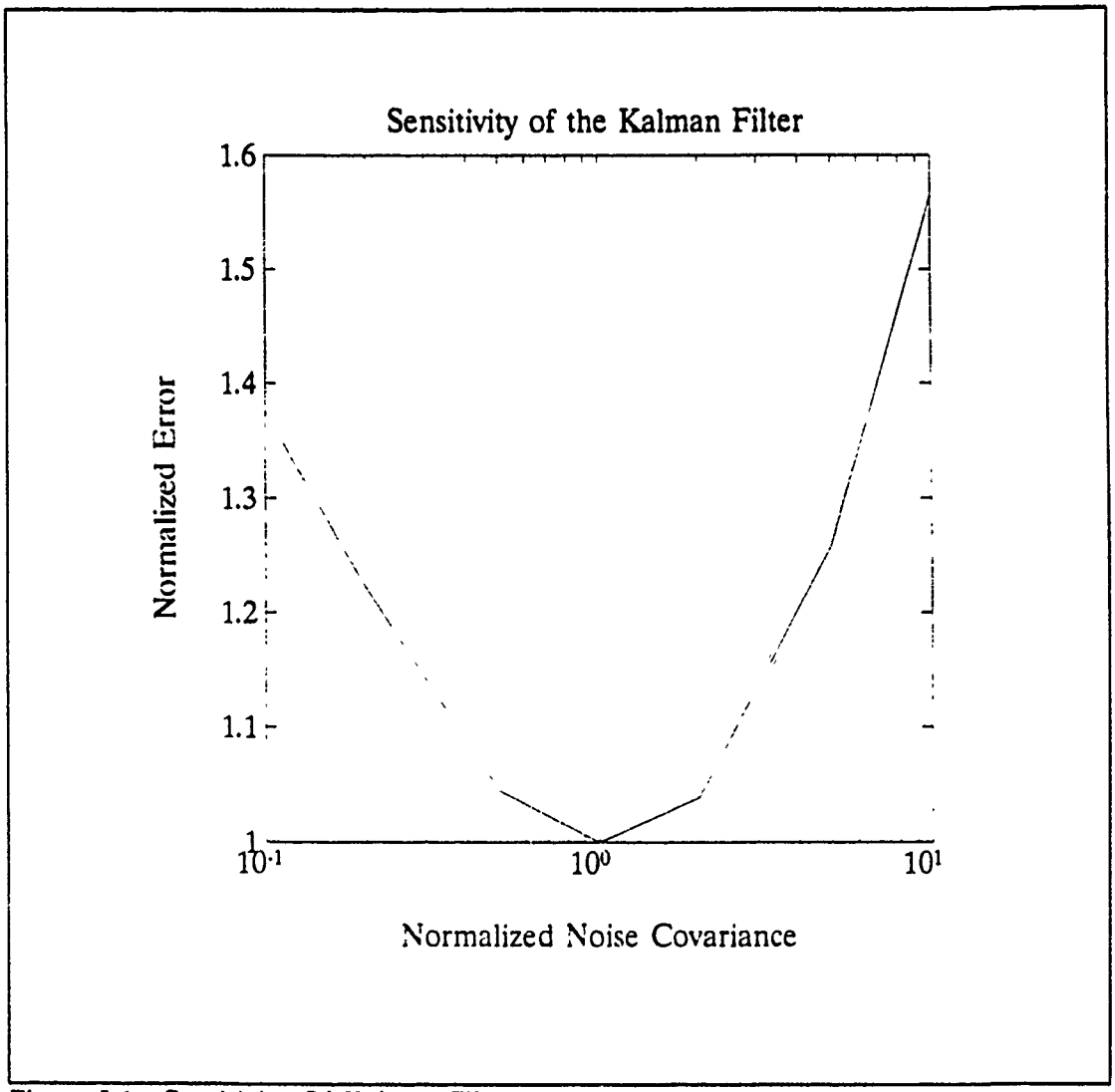

Figure 5.1: Sensitivity Of Kalman Filter

One way of looking at the operation of the Kalman filter is to consider the observations as being a combination of information and noise. The Kalman filter is then a device which extracts the information from the observations and rejects the noise. The Kalman gains are then seen to be directly related to the information in the

observations and inversely related to the noise. As the information goes up, the gains go up; as the noise goes up, the gains go down. The goal of the Kalman equations is to set the gains to maximize the information taken from the observations while minimizing the effects of the noise. This is, of course, predicated on the *a priori* values for the noise that we give to the Kalman filter.

If the *a priori* value of R is larger than the actual ($K_R > 1$) then the Kalman gains will be lower than optimal and the filter will reject some of the information in the observations along with the noise. If the *a priori* value is too small, then the gains will be too large and the filter will treat some of the noise as information. The Kalman is thus dependent upon our choice of *a priori* values of R and requires them to be accurate for its estimates to be optimal.

To satisfy this requirement we must give the Kalman filter either perfect *a priori* noise information, or else the ability to estimate and adapt to the actual value of R. Since perfect *a priori* information is only a concept (as opposed to reality), we have chosen to implement a method which will allow the Kalman filter to adapt to the actual values of the noise.

B. ESTIMATING OBSERVATION NOISE

Before the Kalman filter can adapt to the actual observation noise covariance it must be able to form an estimate of the matrix R. Since the only information available to the Kalman filter comes from the observations, it must calculate an estimate of R as a function of the observations. We will define the estimate of R for observations from time k = n through k = m as

$$\hat{R}_{n,m} = f(z_n, z_{n+1}, \cdots, z_m). \tag{5.7}$$

Since the observations for our system are given as

$$z_{k+1} = H(x_{k+1}) + v_{k+1} \tag{5.8}$$

we cannot directly separate the noise from the observation without an exact knowledge of the state, $x_{k+1|k}$. But if we had this knowledge, there would be no need to use a Kalman filter. The best we can do is to use the residual, r_{k+1}, which provides an estimate of the observation noise.

$$r_{k+1} = z_{k+1} - H(\hat{x}_{k+1|k}) \tag{5.9}$$

This makes our estimate of R in Equation (5.7) a function of both the observations and the state estimates.

$$\hat{R}_{n,m} = f(z_n,\cdots,z_m,\hat{x}_{n|n-1},\cdots,\hat{x}_{m|m-1}). \tag{5.10}$$

Since the state estimates are functions of the observations it might seem unnecessary to use them in Equation (5.10). We do this, however, because the state estimates are also functions of other parameters, notably the *a priori* value of R. So we expect different *a priori* values of R to produce different state estimates for the same set of observations. This is exactly what Figure (5.1) demonstrates. We will now present two different functions for Equation (5.10) which can prcduce estimates of R.

1. Recursive Variance

The most direct (and least memory intensive) method of calculating an estimate of the noise variance is to assume that R is a diagonal matrix (no cross-correlation terms) and to recursively calculate the variance of each component. This

only works if we assume that the residual and the observation noise are both zero-mean. The function which calculates the recursive variance is given as

$$V[i]_{k+1} = \frac{k}{k+1}V[i]_k + \frac{1}{k+1}(r[i]_{k+1})^2 \qquad k = 1...n \qquad (5.11)$$

where the index i spans the number of elements in the residual vector. We then apply Equation (5.11) to each element of the residual vector independently.

This method has the advantages of simplicity and ease of implementation. Its two disadvantages are that it requires many (30 to 40) observations before its estimates become meaningful and that it ignores any possible cross-covariances of the noise terms. The former is rooted in stochastic theory and cannot be circumvented; the latter is solved by the following method.

2. Time Series Covariance

Although we have assumed that the noise components were uncorrelated to simplify the derivations, we will now attempt to estimate the possibly non-zero off-diagonal terms of the matrix R. In order to do this we will use a batch processing function which uses all of the residuals at once, instead of one at a time, as the recursive method does. This function requires a time series of residuals as its argument,

$$\hat{R}_{n,m} = \frac{1}{m-n}\sum_{k=n}^{m}(z_k - \bar{z}_{n,m})(z_k - \bar{z}_{n,m})^T \qquad (5.12)$$

where the mean of the residuals is given by

$$\bar{z}_{n,m} = \frac{1}{m-n+1}\sum_{k=n}^{m} z_k. \qquad (5.13)$$

Equations (5.12) and (5.13) are defined as the sample variance and the sample mean of the parameter z_k.

It is interesting to note that the Kalman filter equations already provide an expected value for the observation covariance matrix which is estimated by Equations (5.12) and (5.13). This value is calculated with each new observation and is the expected covariance of the residual,

$$V[r_{k+1}] = H'P_{k+1|k}H'^T + R_{a\ priori} \tag{5.14}$$

Equating this to our estimates we find that

$$E[\hat{R}_{l,m}] = H'P_{m|m-1}H'^T + R_{a\ priori} \tag{5.15}$$

Solving Equation (5.15) for the value of $R_{a\ priori}$ gives us

$$R_{a\ priori} = E[\hat{R}_{l,m}] - H'P_{m|m-1}H'^T \tag{5.16}$$

which could be used as an estimate for the actual value of R. The problem with Equation (5.16), though, is that the difference of two positive semi-definite matrices is not necessarily another positive semi-definite matrix. Since we require R to be positive semi-definite (noise cannot have negative power), the use of Equation (5.16) can, and has. produced calculational singularities in the Kalman equations.

Although this singularity problem can be controlled by additional processing, we have chosen to implement Equation (5.12) as our estimate of the actual noise covariance. It is more computationally stable than Equation (5.16), more robust than (5.11), and provides an estimate which is conservative; the expected value of the estimate is larger than the actual values, as shown by Equation (5.16). Now that we have chosen

a method for estimating the actual observation noise covariance matrix, R, we will present a method for incorporating this information into the Kalman filter.

C. NOISE ADAPTATION

The purpose of estimating the observation noise covariance is to provide the Kalman filter with the ability to both correct for poor *a priori* estimates and to adapt to non-stationary noise processes. Since the primary objective of Kalman filtering is to provide the best state estimates possible, we will not allow our design to be driven by dogmatic concerns for computational efficiency. It is expected that an adaptive process will require more processing, and we will suffer this, within limits. We will also consent to the use of more memory for the computations. Because these issues of speed of memory are being overtaken by technology, we will stretch our design to maximize the benefits at a moderate expense of processing and memory.

With this license to moderately increase processing and memory requirements, we have chosen to implement the observation noise adaptation in the form of a smoothing process. This means that the Kalman filter carries along a finite, growing memory which stores the observations and the residuals. At specified intervals the estimate of R is calculated from the collected residuals. This new estimate of R is used to recompute the estimates from the stored observations. The filter then releases the stored residuals and observations and the process repeats.

This explanation is best served by an example. Table (5.1) details the operation of a Kalman filter which adapts every 30 observations. The number of observations between adaptations should be chosen on the basis of a Monte Carlo analysis for each specific problem. Although this number will vary, we have achieved consistently good

results with a value of 30. This is a reasonable value as it does not require much increase in either processing speed or memory.

TABLE 5.1: EXAMPLE ADAPTATION ALGORITHM

Time index	Action	Store
1	Compute $\hat{x}_{1\|0}$	z_1, r_1
2	Compute $\hat{x}_{2\|1}$	z_2, r_2
:	:	:
30	Compute $\hat{x}_{30\|29}$	z_{30}, r_{30}
30	Compute $\hat{R}_{1,30}$	
30	Recompute $\hat{x}_{1\|0}$ through $\hat{x}_{30\|29}$	Release memory
31	Compute $\hat{x}_{31\|30}$	z_{31}, r_{31}
32	Compute $\hat{x}_{32\|31}$	z_{32}, r_{32}
:	:	:
60	Compute $\hat{x}_{60\|59}$	z_{60}, r_{60}
60	Compute $\hat{R}_{31,60}$	
60	Recompute $\hat{x}_{31\|30}$ through $\hat{x}_{60\|59}$	Release memory

The iterative algorithm shown in Table (5.1) can be implemented as long as the calculations needed to recompute the state estimates can be completed between observations. For our simulations this is not a problem as the observations occur only a few times per minute.

Armed with this method of noise adaptation, we will now investigate and derive a method for allowing the Kalman filter to adapt to a maneuvering target. Following that, we will present the simulation results which show the benefits of this adaptation scheme.

VI. MANEUVER GATING

This chapter introduces the concept of correlated maneuver gating. It begins by defining the Mahalanobis Distance and explains how it can be used as a test statistic for maneuver gating off of the residual. We then explain how the Kalman filter responds to a detected maneuver by adapting its error covariance matrix. This adaptation is usually in the form of a reset to an initial value, but we present a more sophisticated method which we call incremental, correlated maneuver gating.

A. RESIDUAL GATING

The first step in correcting for improper behavior in a Kalman filter is detecting that the behavior has occurred. This is the goal of maneuver gating. Once we have achieved a method which has both a high probability of detection (detecting maneuvers) and a low probability of false alarm (rejecting noise), then we will employ a method to correct for the filter's behavior.

The method of adapting a Kalman filter by maneuver gating has been used widely in the literature in the past several years. In every case the test statistic used to determine the gate was one or more elements of the residual, which was compared to some number (the gate). Often the gate size was determined by analysis of the covariance of the residual, the very method we shall use. What we propose, however, is that the entire residual be compared to the covariance of the residual by calculating the Mahalanobis Distance of the residual.

The Mahalanobis Distance, MD, of a random number is its distance from the sample mean in units of the sample covariance. This is given by

$$MD = (X - M)^T K^{-1} (X - M) \qquad (6.1)$$

where X is the random vector, M is the mean of the stochastic process, and K is its covariance. Since we are calculating this statistic for the residual, we make the assumption that the mean of the residual is zero. If we know that the observations have a predetermined bias, we would compensate for it so that the residual would still be zero-mean.

Since the noise processes involved in our system lead to residuals with a zero-mean Gaussian distribution (as shown in Appendix A), we see that each value of the Mahalanobis Distance for the residual describes a two-dimensional ellipse around the state estimate projection. The size of the ellipse is directly proportional to the Mahalanobis Distance. This is seen graphically in Figure (6.1). By recalling that the covariance of the residual is given by

$$V[r_{i|k+1}] = H'P_{k+1|k}H'^{T} + R \tag{6.2}$$

we see that the covariance of the residual is a strong function of the observation noise covariance. In fact, as the error covariance decreases, the first term in Equation (6.2) becomes negligible and the observation noise covariance, R, dominates. So the process of maneuver gating reduces to the problem of discriminating between noise in the observations and a target maneuver. This will be covered more thoroughly in the last section of this chapter.

The determination of the optimal value for the Mahalanobis Distance for the gate should balance the need for early maneuver detection against the probability for false detections. This value needs to be determined empirically for each implementation of the Kalman filter and will be shown in the following chapter to be dependent upon the target's actions.

The set of observations and estimates in Figure (6.1) demonstrates the action of the Kalman filter when no maneuver gating is used. As seen, the estimates diverge from the observations because the filter gains have decreased below the value needed to follow the observations. This is a direct result of the Kalman filter's error covariance matrix decreasing in magnitude. In order to allow the Kalman filter to adapt to a maneuvering target we will dynamically adjust the magnitude of its error covariance matrix. In the next section we describe the widely used method of error covariance reset as well as propose a method which we call incremental, correlated maneuver gating.

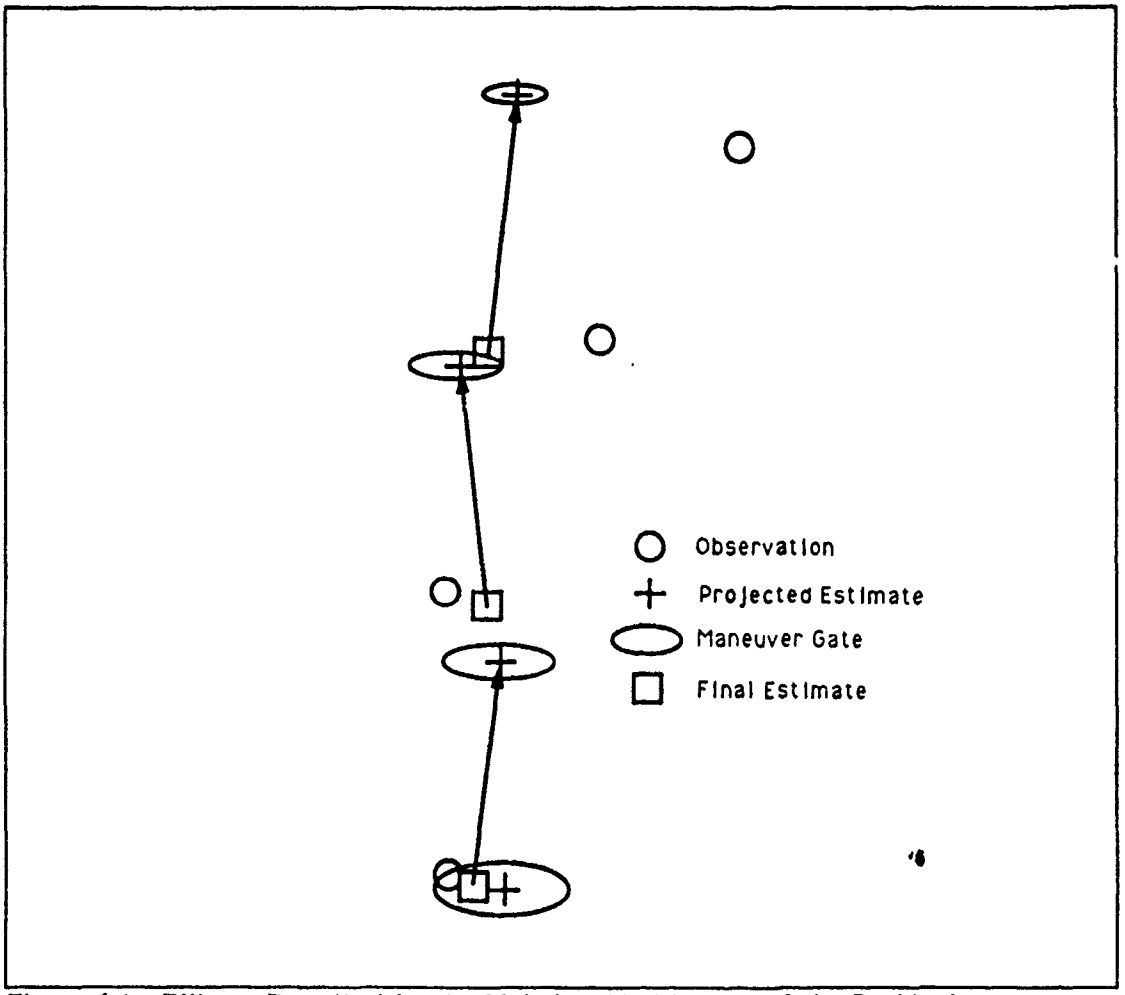

Figure 6.1: Ellipses Described by the Mahalanobis Distance of the Residual (No Gating)

B. ERROR COVARIANCE ADJUSTMENT

In Chapters II and III we showed that the Kalman filter gains are a function of the state error covariance. As the estimated error goes up, so do the gains. This is the filter's way of compensating for a large expected error; it weights the observations more heavily. We can exploit this behavior of the Kalman filter and use it to adapt the Kalman filter to detected target maneuvers. Whenever we detect a target maneuver, we increase the magnitude of the error covariance matrix. This forces the Kalman filter to move its state estimate update closer to the observation. The question remains, how much do we adjust the error covariance and how close to the observation do we want the estimate to be?

1. Covariance Reset

The commonly used method of error covariance adjustment is to reset the error covariance to its initial value ($P_{0|0}$). This initial value is simply the value set by the filter designer which allows the filter to lock-on to the first observation. Since this value is typically large compared to the steady-state value, the Kalman filter will behave as if it has been reinitialized whenever a maneuver is detected.

This type of covariance adjustment will force the Kalman filter to disregard all past information and reinitialize itself on the current observation. This behavior will be destructive when the gating is a result of anomalous observation noise rather than target maneuvering. The Kalman's state estimate will simply bounce around chasing the noisy observations and will result in a larger state estimate error. In addition, the large magnitude of the initial error covariance matrix will require several time intervals to again reach a steady-state value. This method of gating can therefore lead to an effect which will require several observations from which to recover. What is desired is a

method of gating which will allow the Kalman filter to "relax" back to a steady-state condition as soon as possible after gating ceases. Such a method is presented below.

2. Covariance Increment

Some of the objectionable, and persistent, effects of covariance reset gating can be negated by increasing the error covariance matrix to a magnitude smaller than its initial value. The determination of this optimal magnitude, however, is not immediately computable from known data because it is an unknown function of the state of the Kalman filter. Its value will therefore have to be determined empirically from experimentation.

Our method for choosing the value of the error covariance matrix consists of incrementally increasing the magnitude of the old covariance matrix until the maneuver gate is satisfied. An example of this is shown in Figure (6.2). Although this method is heuristic in nature, we have achieved consistently good results with it, as the simulations in Chapter VII will show.

C. CORRELATED GATING

The plight of reset gating, as described above, is its vulnerability to aberrant noise conditions. To counter this weakness, the maneuver detection algorithm must make use of the statistical properties of the noise to distinguish the noise from a maneuver. Since we assume that the noise is zero-mean, additive, white, Gaussian noise (AWGN), we will use the property of whiteness and the mean as test statistics for our comparison. The following sections describe simple methods of exploiting these noise properties for the purpose of maneuver detection.

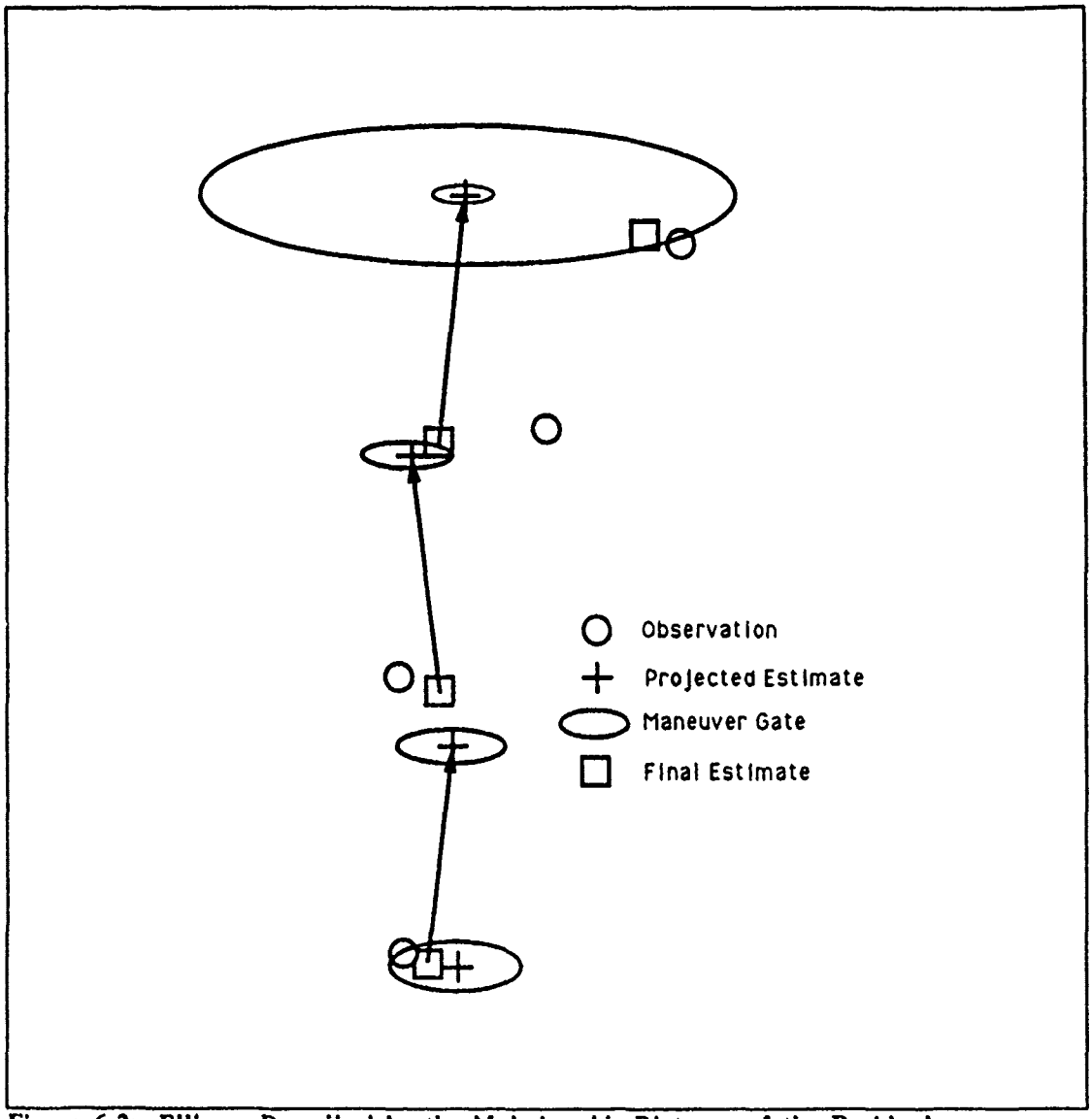

Figure 6.2: Ellipses Described by the Mahalanobis Distance of the Residual (Incremental Gating)

1. Temporal Correlation

The property of whiteness of a random process requires that the spectral distribution of that process be flat. Whiteness also requires that samples of the process taken at different times be uncorrelated. For Gaussian processes, this leads directly to

the property that samples taken at different times are independent. Simply put, the value of the noise at a given time is independent of the value of the noise at any other time.

But the same is not true for a target maneuver. When the target accelerates to one side, we expect that the observations will continue to be off to that side until the Kalman filter recaptures the target. Even then, the observations will again drift off to the side until the target stops accelerating. So the time during which the observations drift due to target maneuvering is strongly correlated.

We can exploit this difference in time correlation by requiring that residuals exceed the maneuver gate more than just once in a row to be considered a maneuver. We chose to require two consecutive gatings to signal a target maneuver. This provided a compromise between fast response and low false alarm rates. On the average, this method would reject 50% of the aberrant noise signals. If we required three consecutive gates, we would reject 75% of the aberrant noise, but the filter response would be slowed. To improve gating performance, we chose to use two consecutive gates and then further process the gate by requiring spatial correlation.

2. Spatial Correlation

Just as the noise values are uncorrelated in time, so are they uncorrelated in space. This is a direct consequence of the noise being AWGN of zero-mean. Likewise, the maneuver observations are heavily correlated in space, and remain so until the maneuver is complete. Using a method similar to that used for temporal correlation, we signal a maneuver only when two consecutive gates are detected on the same side of the projected observation. This method further reduces the false alarm rate by the same

order as that of temporal gating. All that is required now is to determine the size of the gate.

As will be shown in Chapter VII, a nominal value for the gate size is two. This value rejects all noise values which are less than two standard deviations in magnitude. Since the noise is Gaussian, this will give an average false alarm rate of only five percent. By further processing with temporal and spatial correlation, the false alarm rate is reduced to about one percent.

VII. SIMULATIONS

This chapter presents only the simulations; the results are discussed in the next chapter. The simulations which are presented in this chapter are in accordance with the scenarios given in Table (4.1) and are given in two sections. The first section shows the results of the noise power estimator described in Chapter V, while the second section gives the results of the maneuver gating schemes described in Chapter VI. These simulations were run using the program code given in Appendix C.

The noise used in the simulations was generated by software which implemented a pseudo-random noise algorithm. This noise sequence was then filtered to reduce the effect of abnormal sequences generated by the algorithm. To further reduce the effects of singularly beneficial (or malevolent) sequences, the simulations were each run for a suite of 30 independent noise sequences. The results given for each scenario is the average of these 30 simulations. For the plots of the target tracks, one noise sequence was chosen at random to be representative of the suite.

A. NOISE ADAPTATION

Figures (7.1) and (7.2) show the output of the noise power estimator as a function of the input observation noise power for the non-maneuvering scenarios, one through three. The noise powers are normalized to the values given in Equations (4.15) and (4.16). Separate graphs are given for the noise power estimates in the bearing and range dimensions.

Using these estimates in the non-maneuvering scenarios results in the performance shown in Figures (7.3) through (7.5). These graphs show the relative errors of the Kalman filters as the observation noise power is scaled from its nominal values.

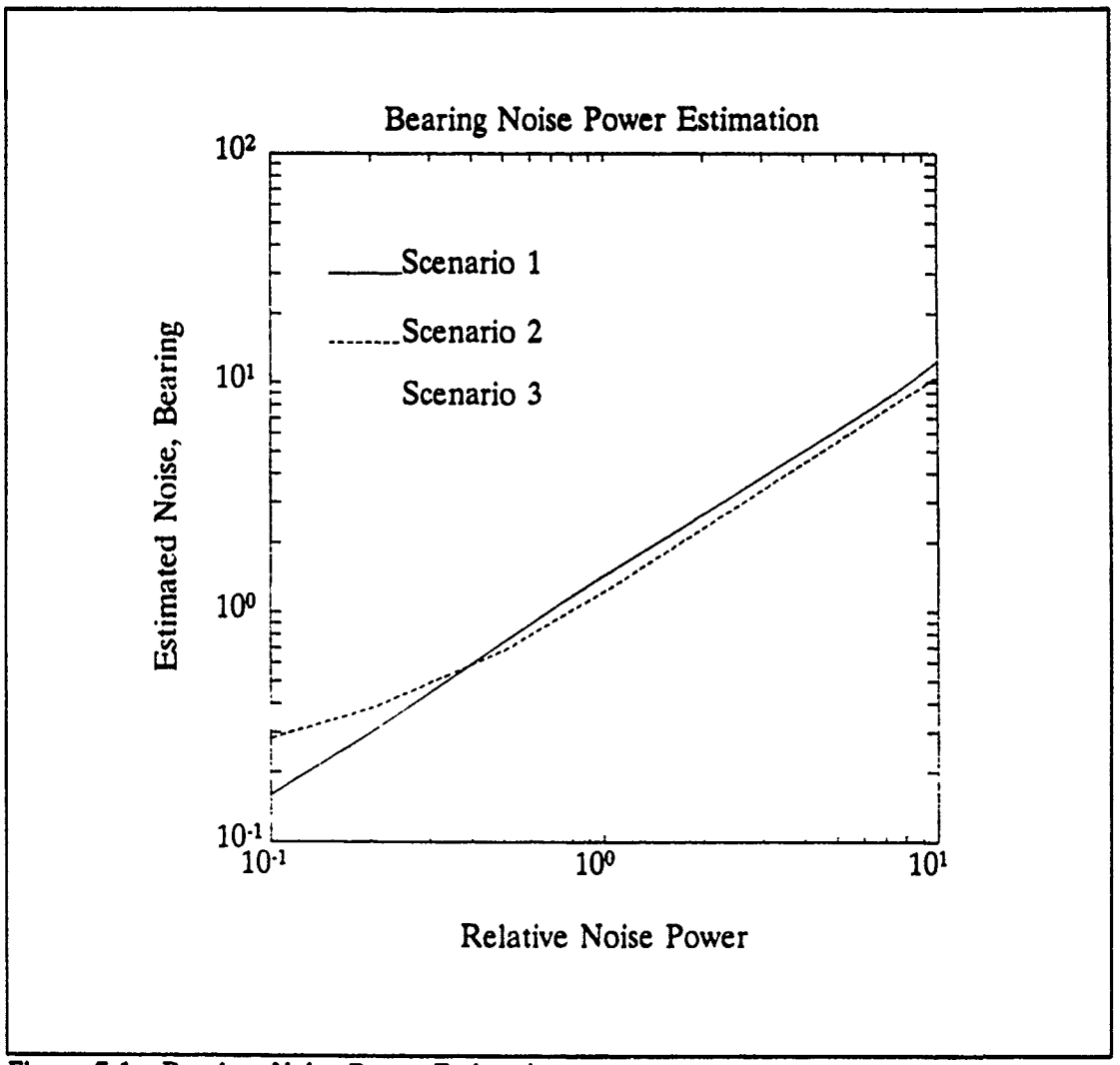

Figure 7.1: Bearing-Noise Power Estimation

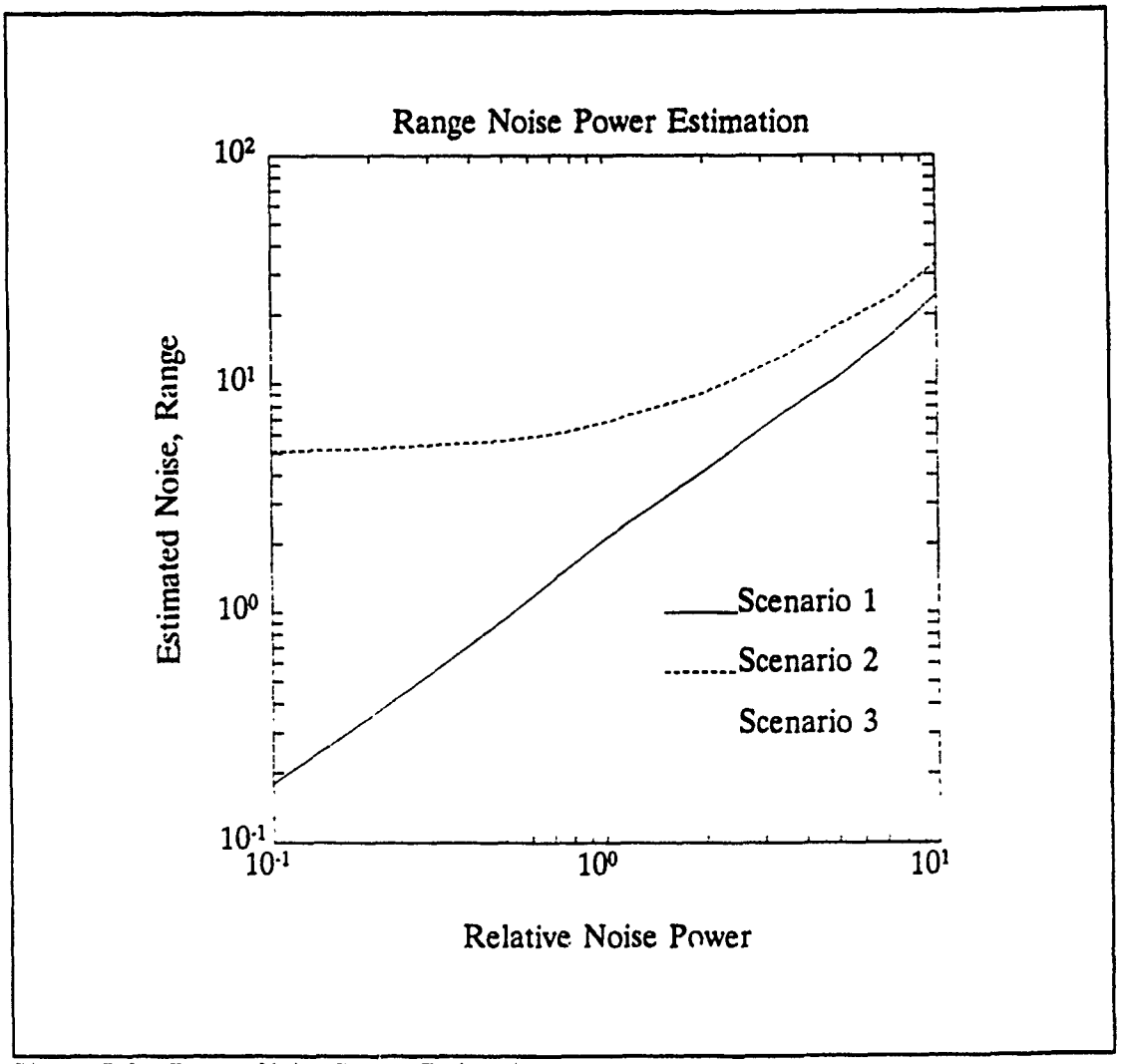

Figure 7.2: Range-Noise Power Estimation

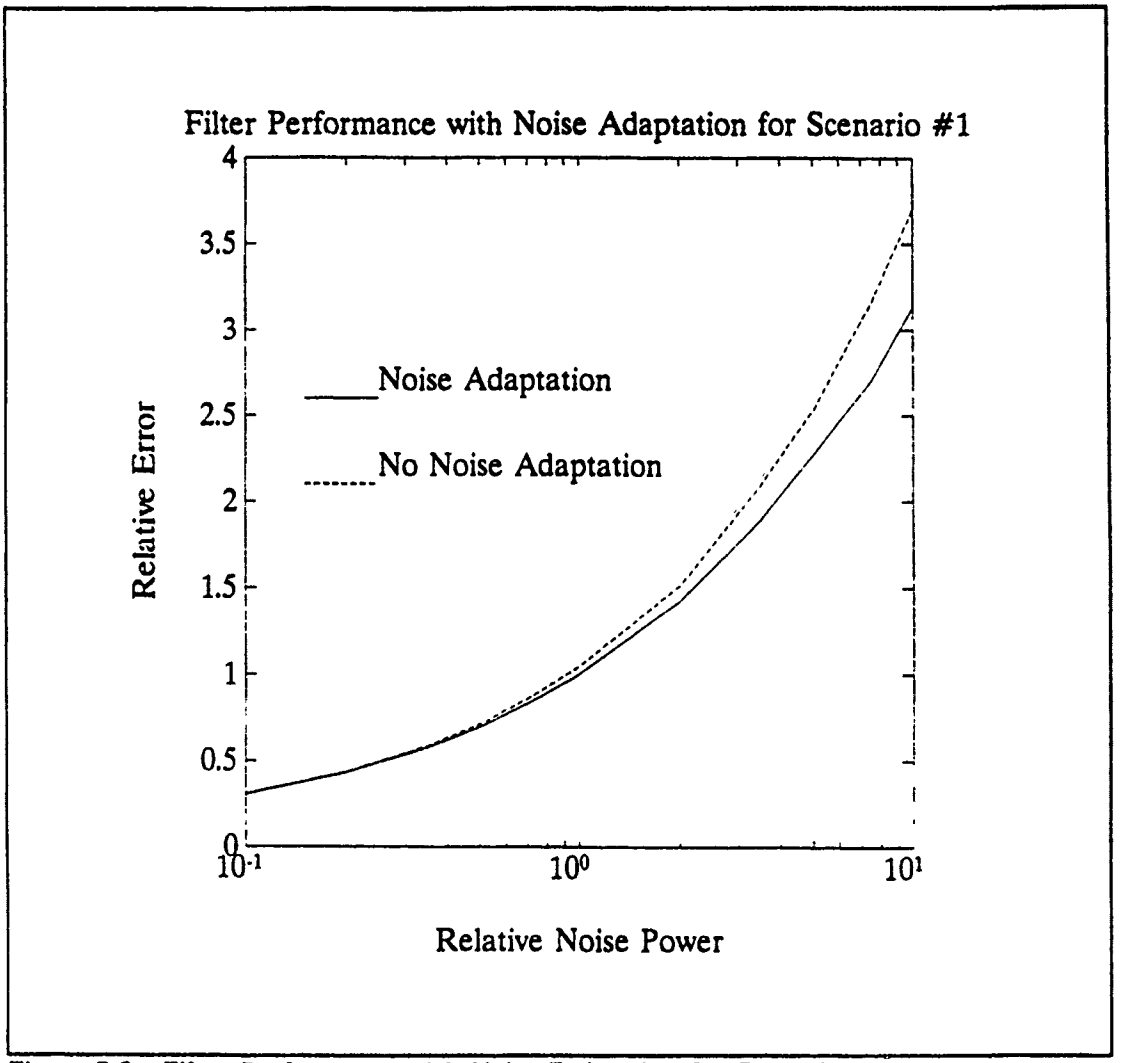

Figure 7.3: Filter Performance with Noise Estimation for Scenario #1

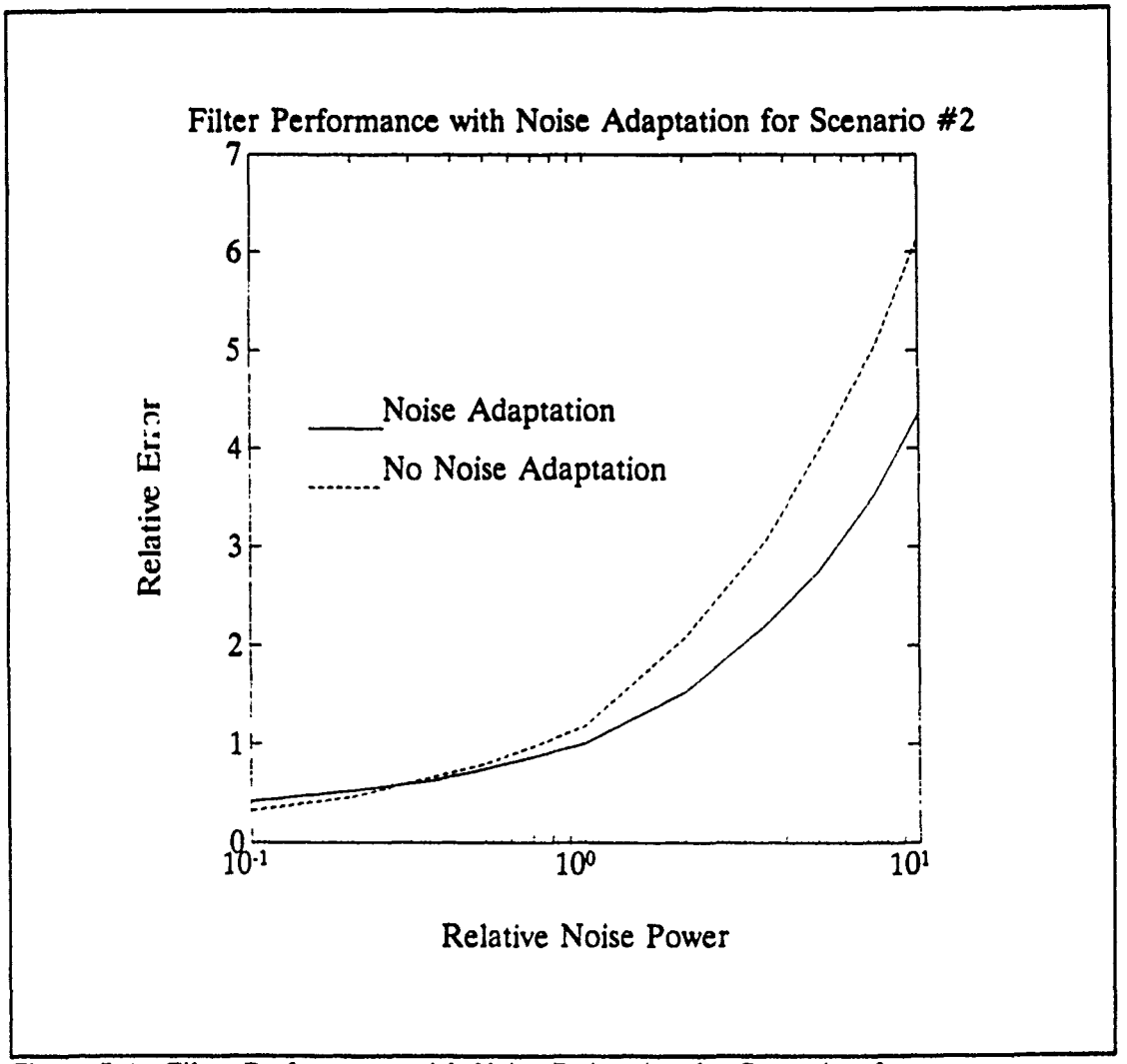

Figure 7.4: Filter Performance with Noise Estimation for Scenario #2

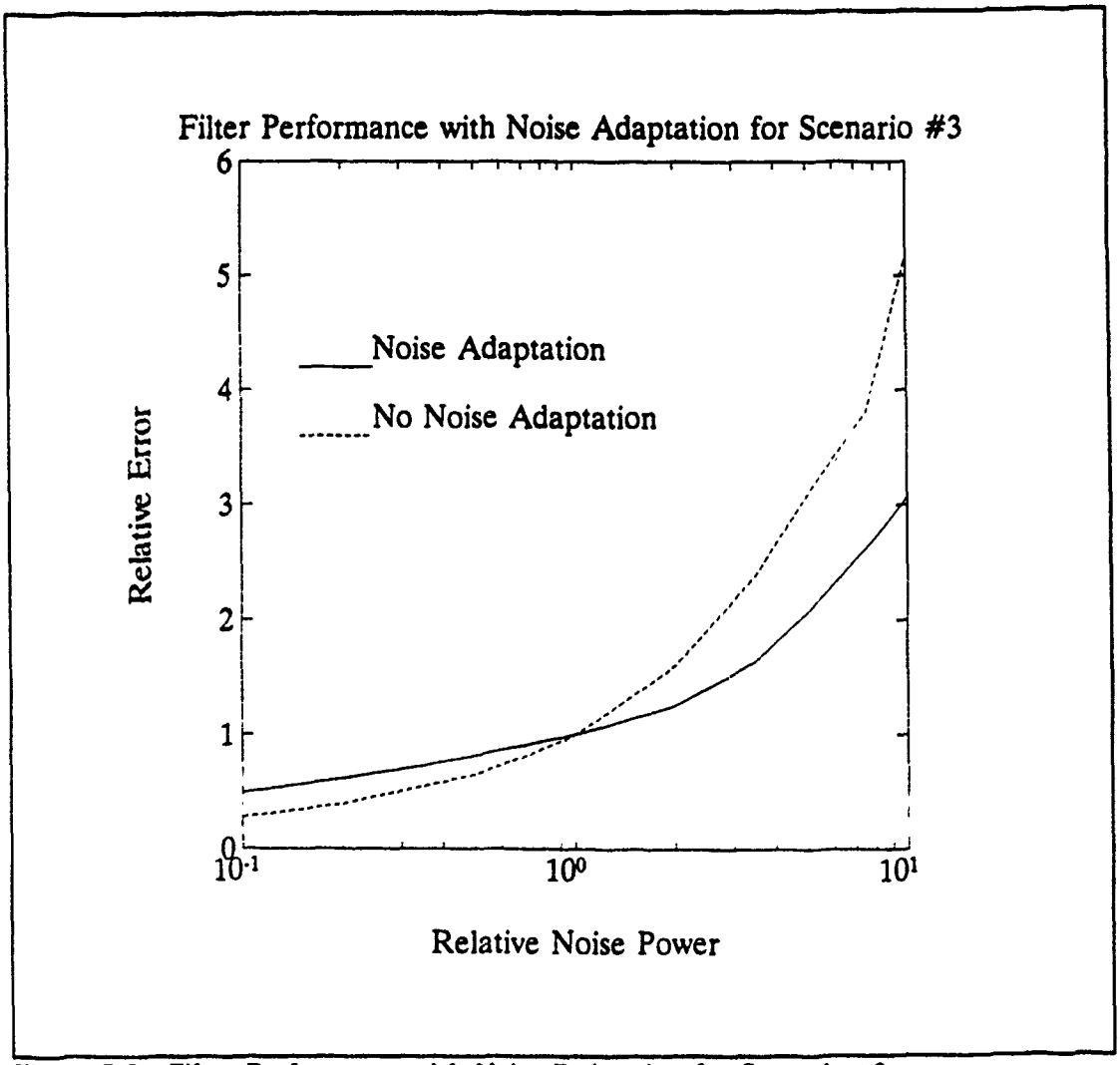

Figure 7.5: Filter Performance with Noise Estimation for Scenario #3

B. MANEUVER GATING

The simulations for the maneuver gating performance are given in five stages. The first stage, Figures (7.6) and (7.7), shows the filter performance as a function of the maneuver gate size for the maneuvering and non-maneuvering scenarios. Using this data, the second stage shows the filter performance for scenarios two through five for each of the gating methods described in Chapter VI. These are shown in Figures (7.8) through (7.11).

The third stage of simulations shows the plot of the target tracks and the filter estimates for a representative noise vector for each of the gating methods using scenarios four and five. These graphs are given in Figures (7.12) through (7.17). The fourth stage, Figures (7.18) through (7.23), shows the actual and expected errors for the graphs given in stage four. The fifth stage compares the actual errors for the two gating methods for the simulations run in stage four, and is shown in Figures (7.24) through (7.27).

The results of these simulations and their implications are discussed in the next chapter. All conclusions are summarized in the final chapter, Chapter IX, along with recommendations for further investigation.

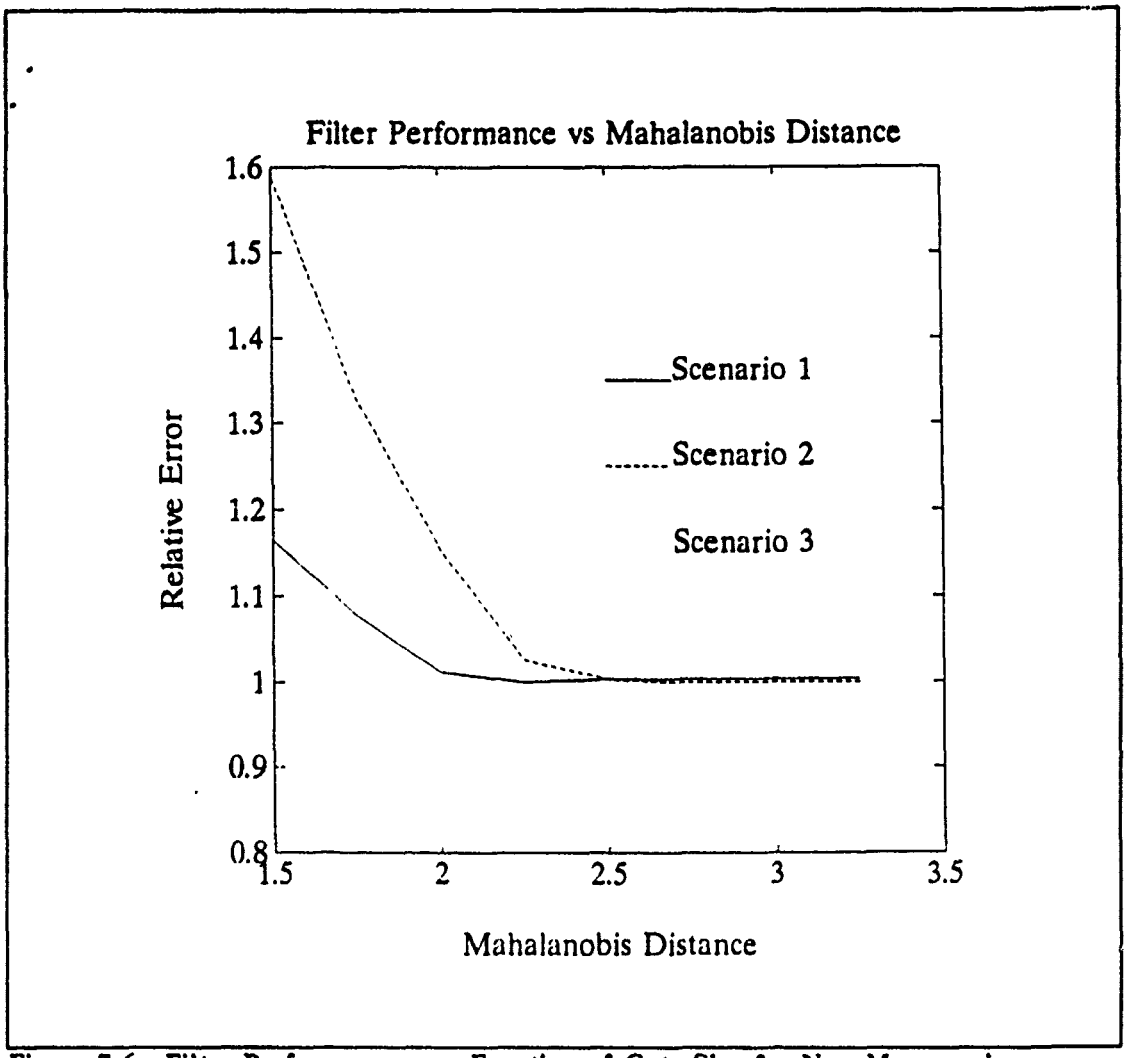

Figure 7.6: Filter Performance as a Function of Gate Size for Non-Maneuvering Targets

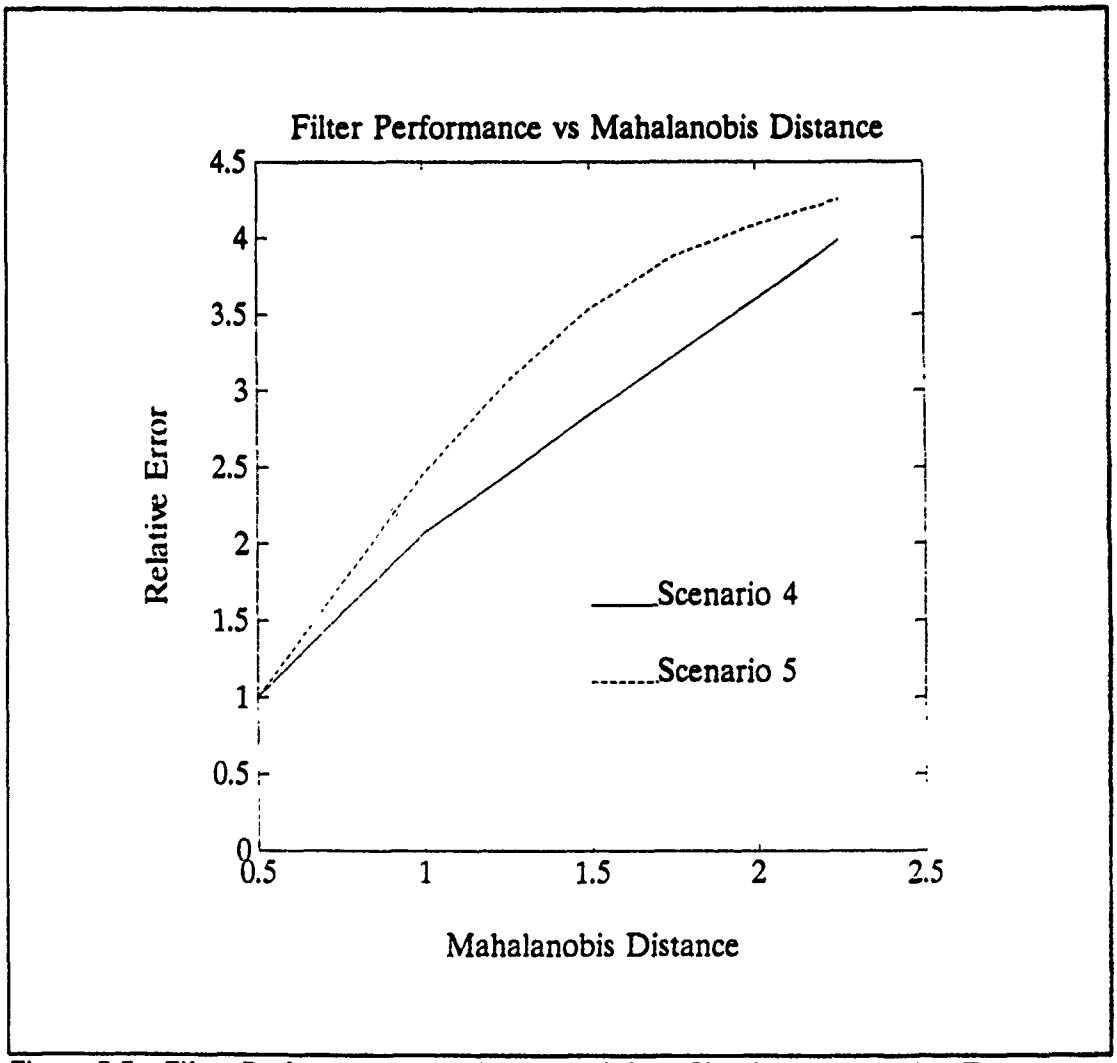

Figure 7.7: Filter Performance as a Function of Gate Size for Maneuvering Targets

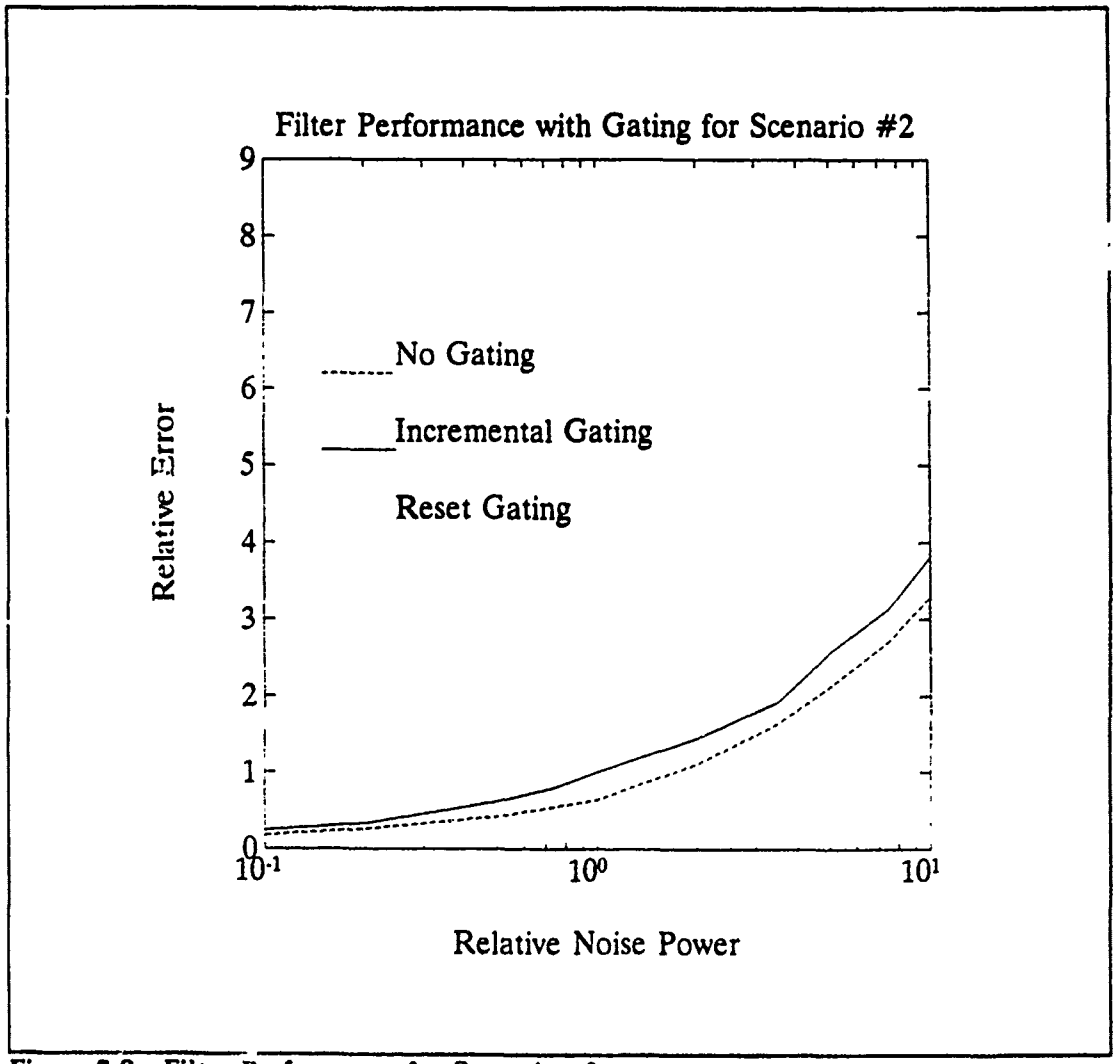

Figure 7.8: Filter Performance for Scenario #2

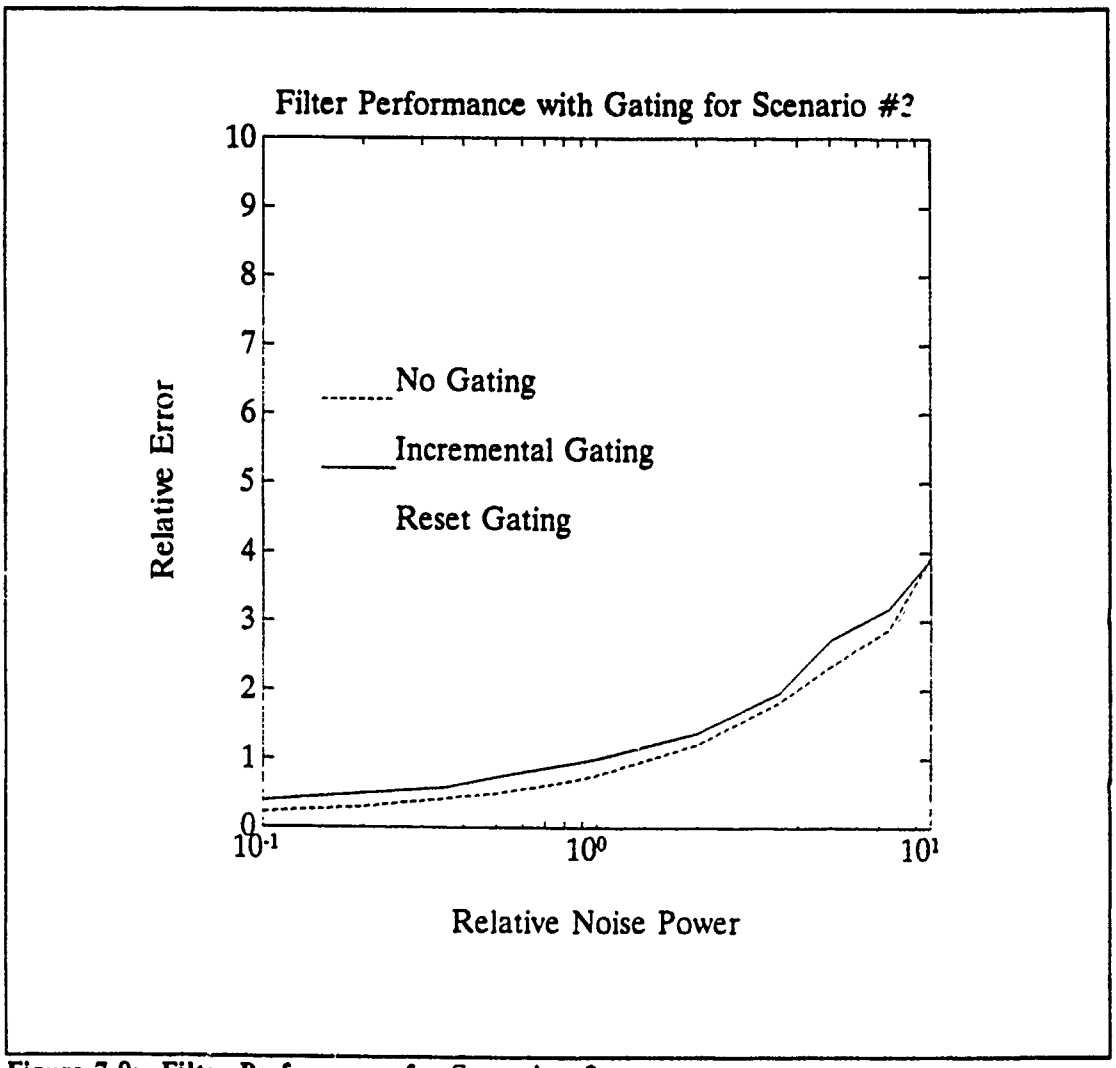

Figure 7.9: Filter Performance for Scenario #3

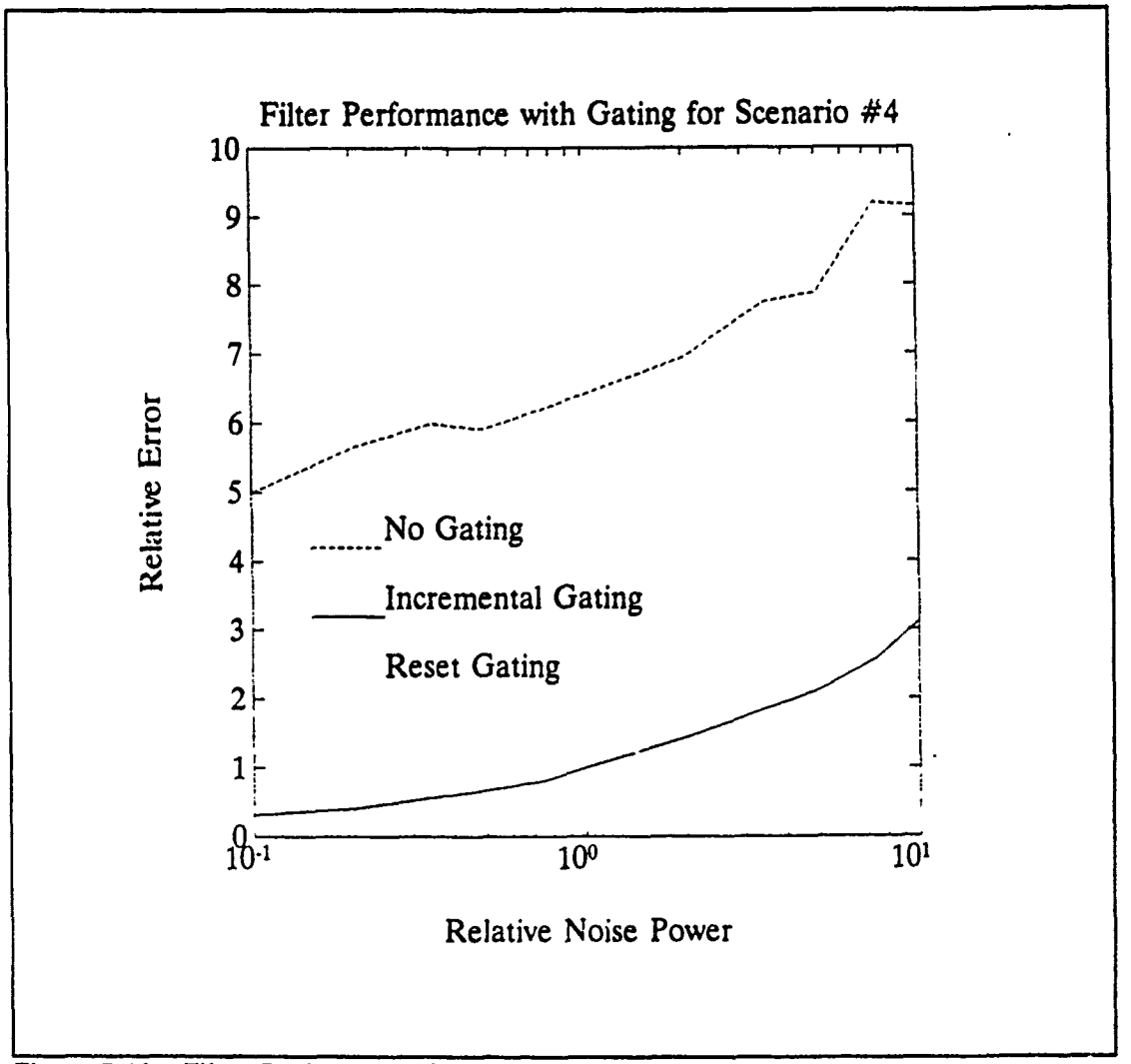

Figure 7.10: Filter Performance for Scenario #4

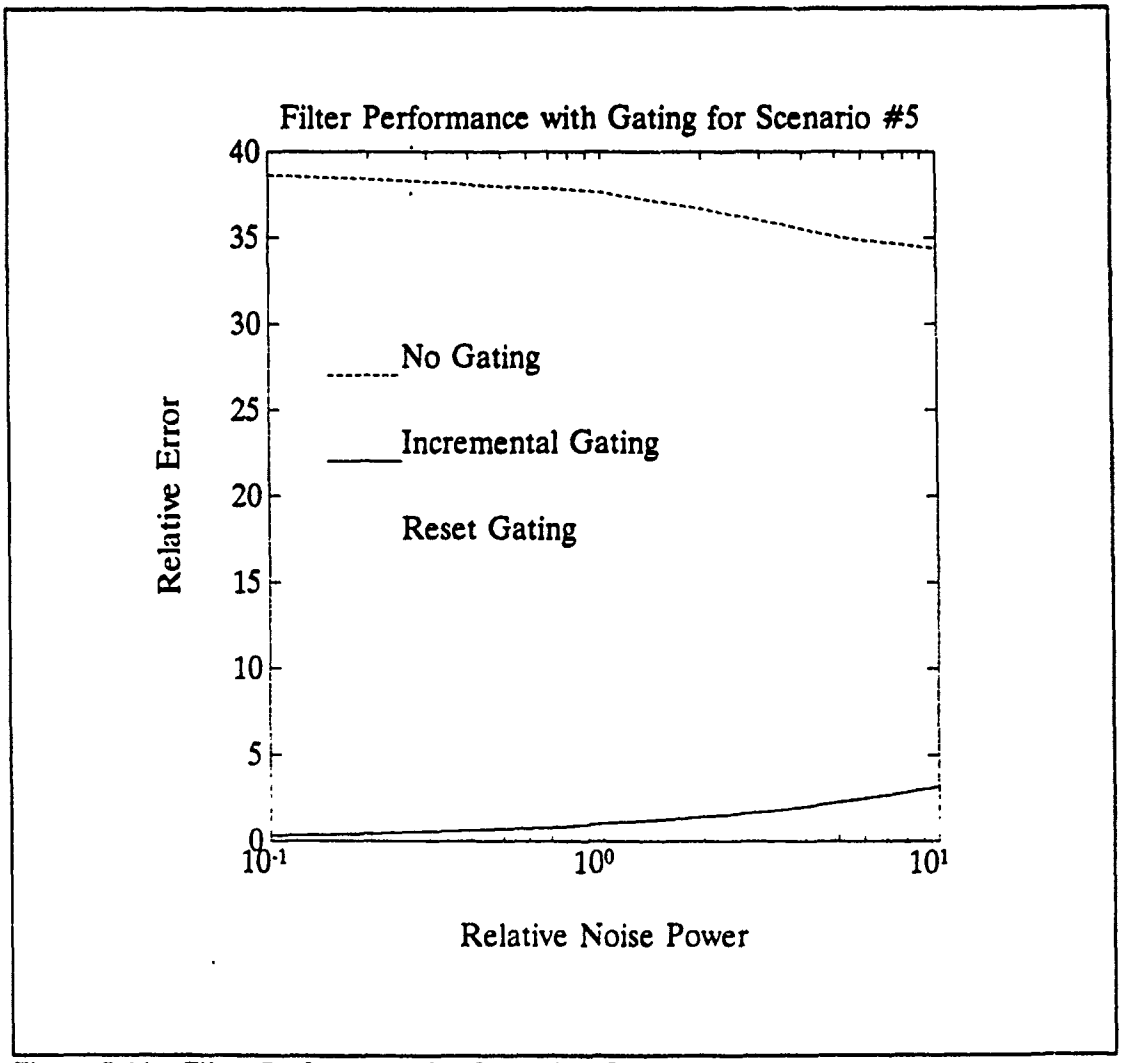

Figure 7.11: Filter Performance for Scenario #5

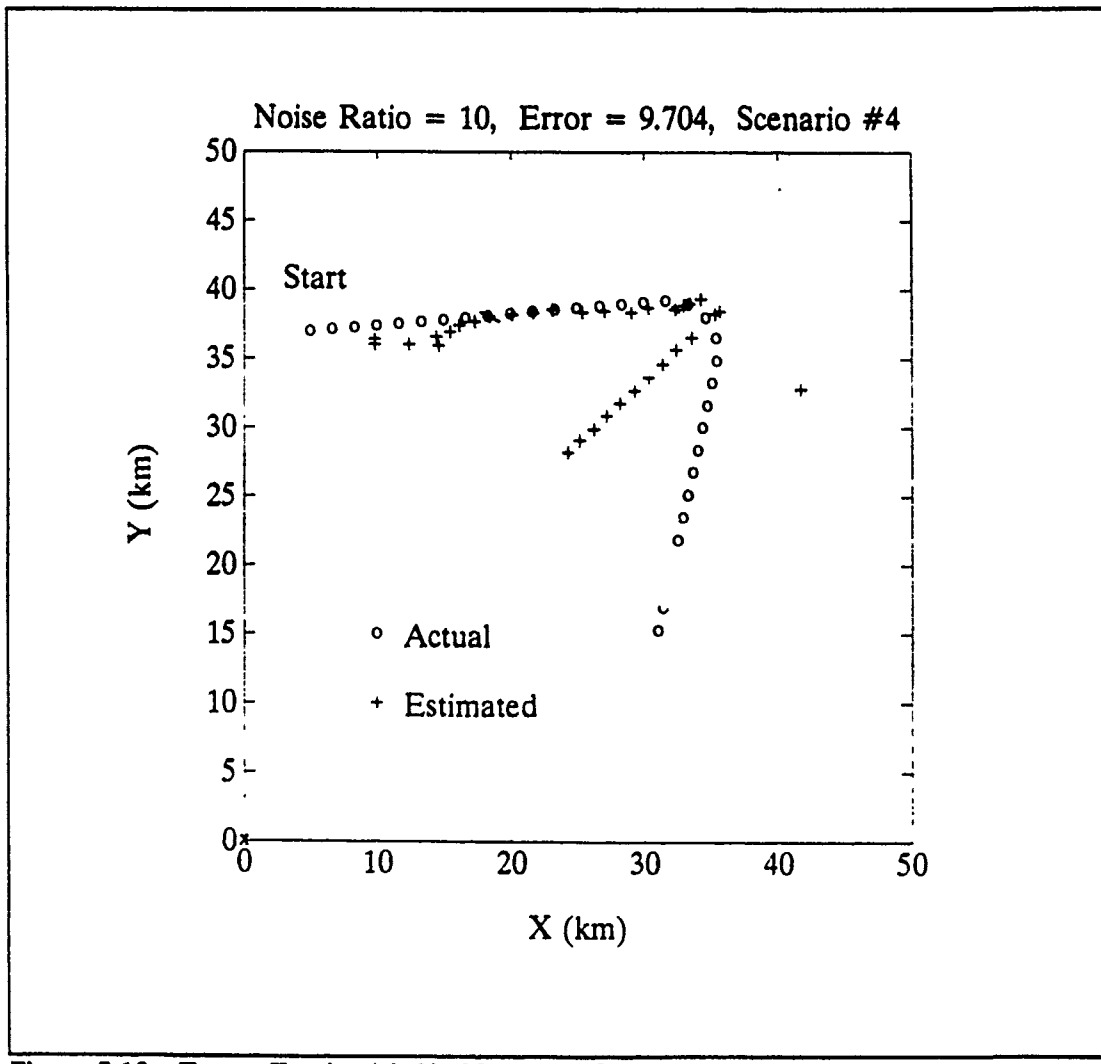

Figure 7.12: Target Track with No Maneuver Gating

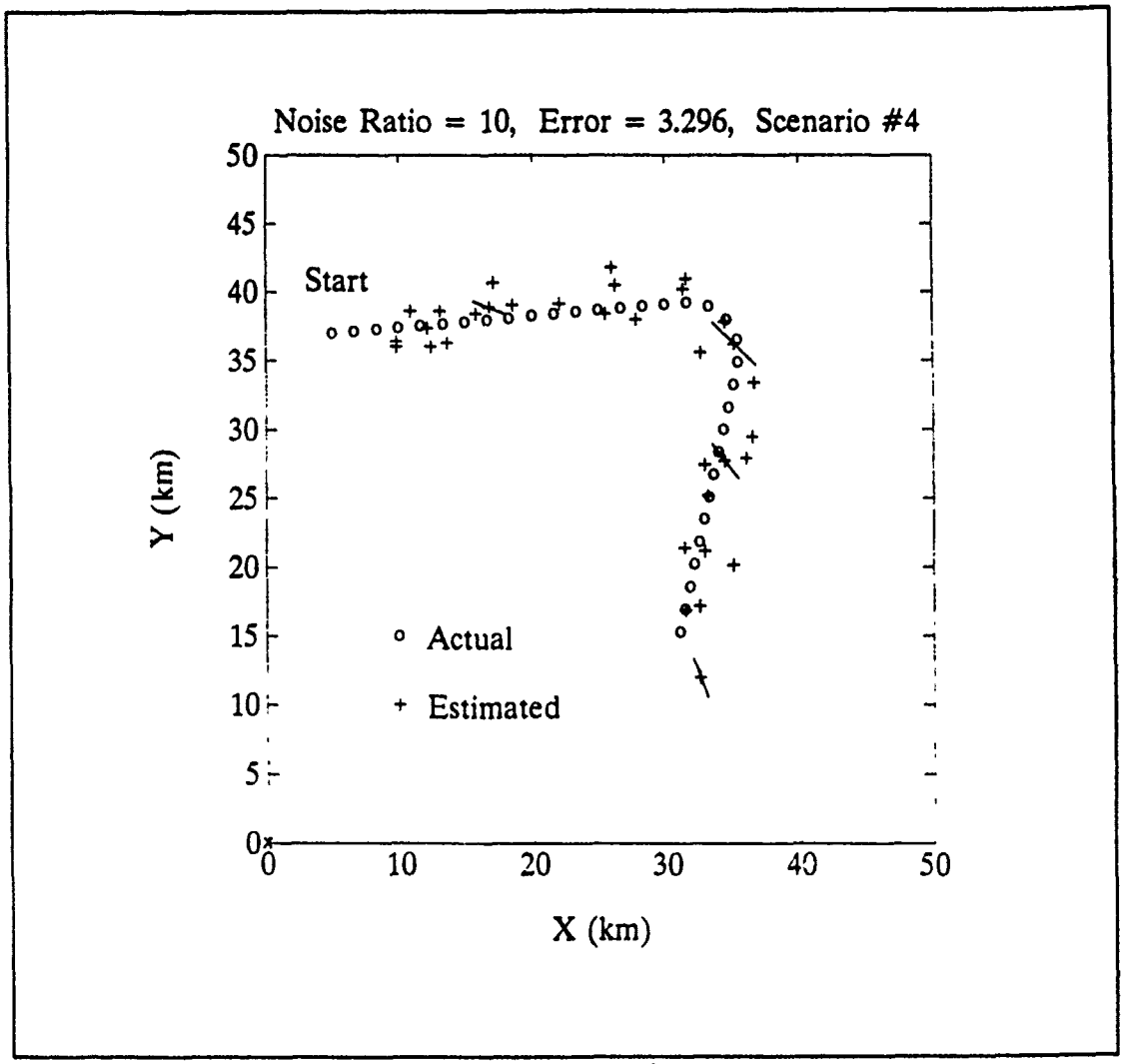

Figure 7.13: Target Track with Reset Covariance Gating

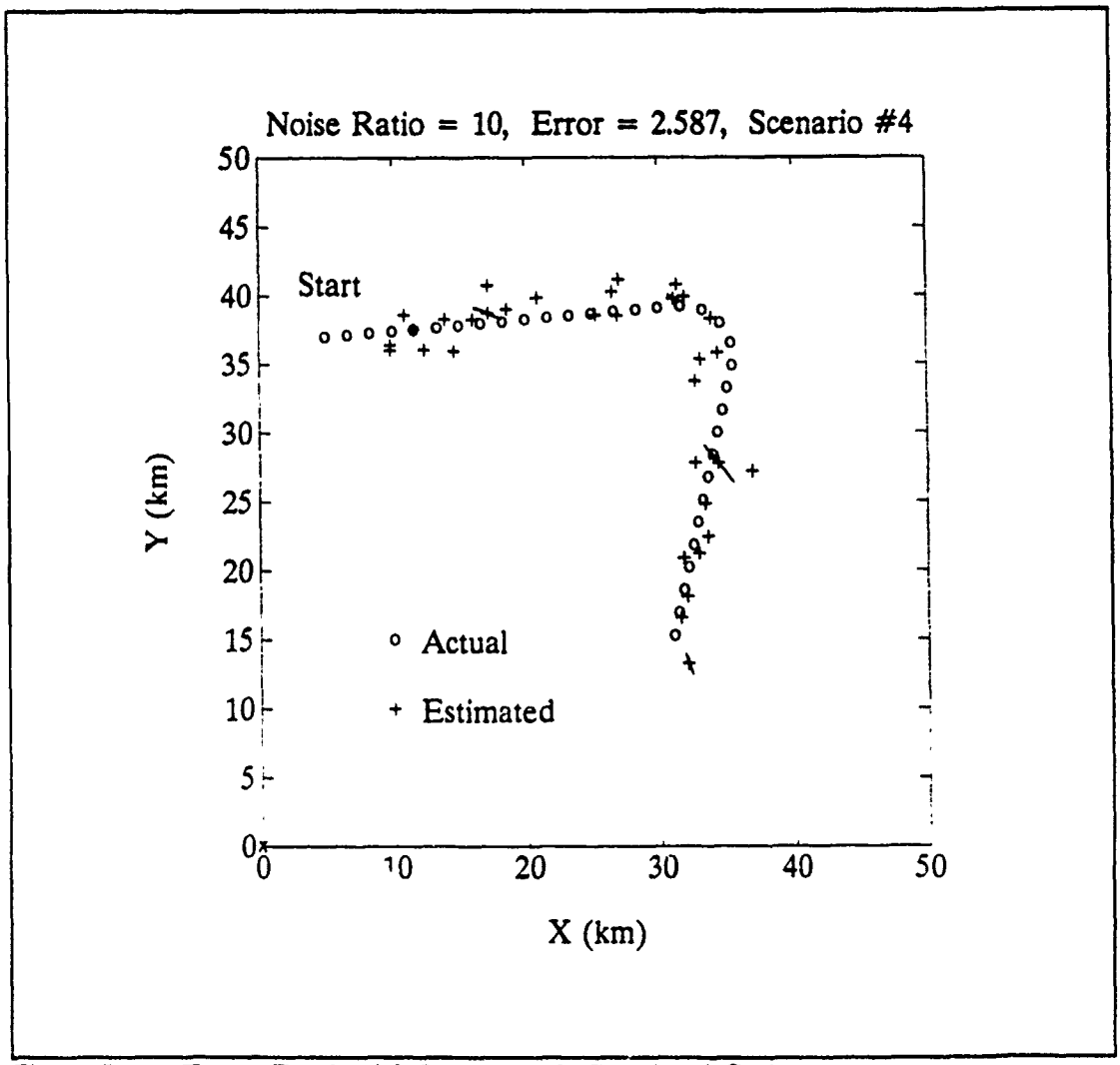

Figure 7.14: Target Track with Incremental, Correlated Gating

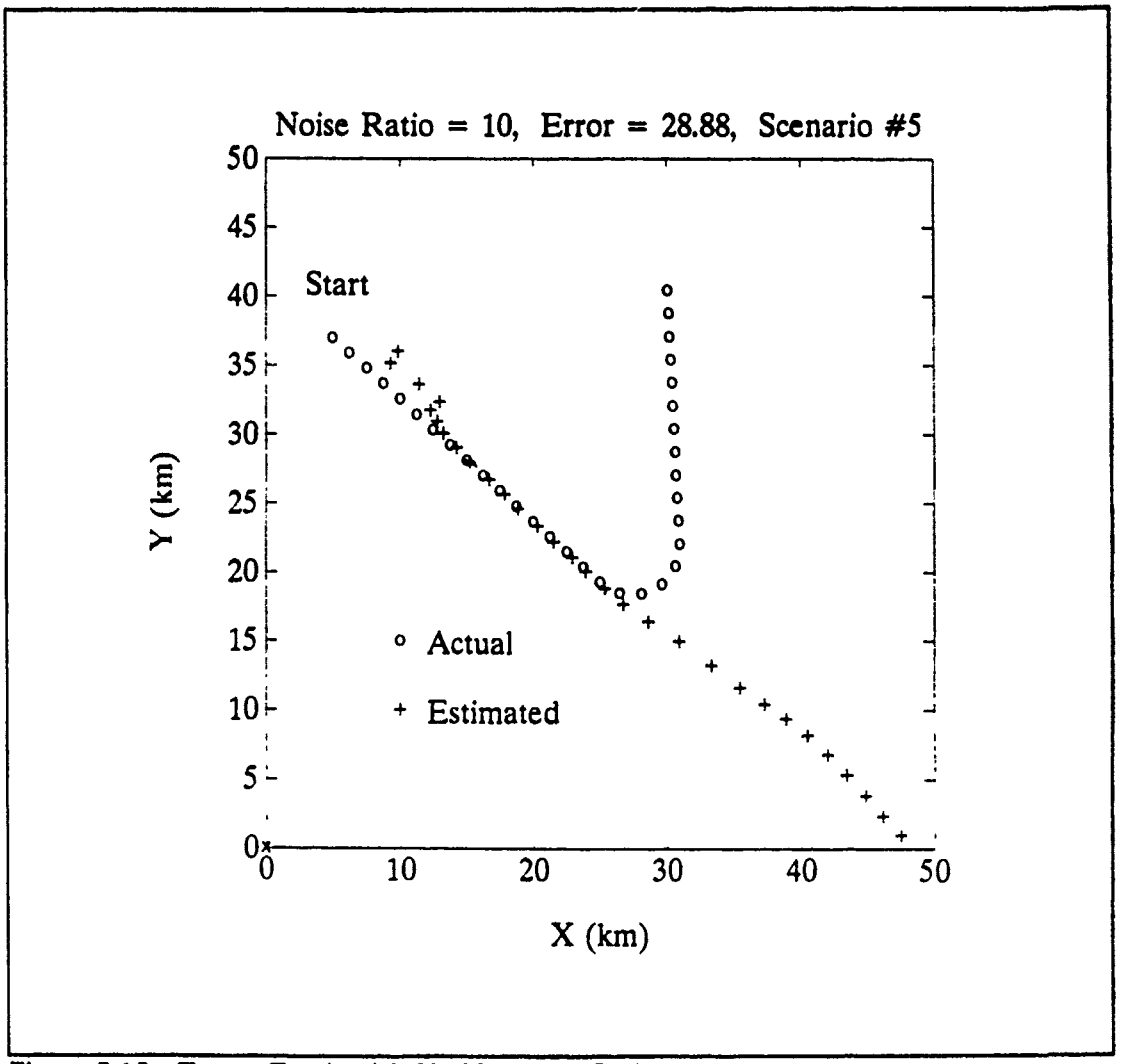

Figure 7.15: Target Track with No Maneuver Gating

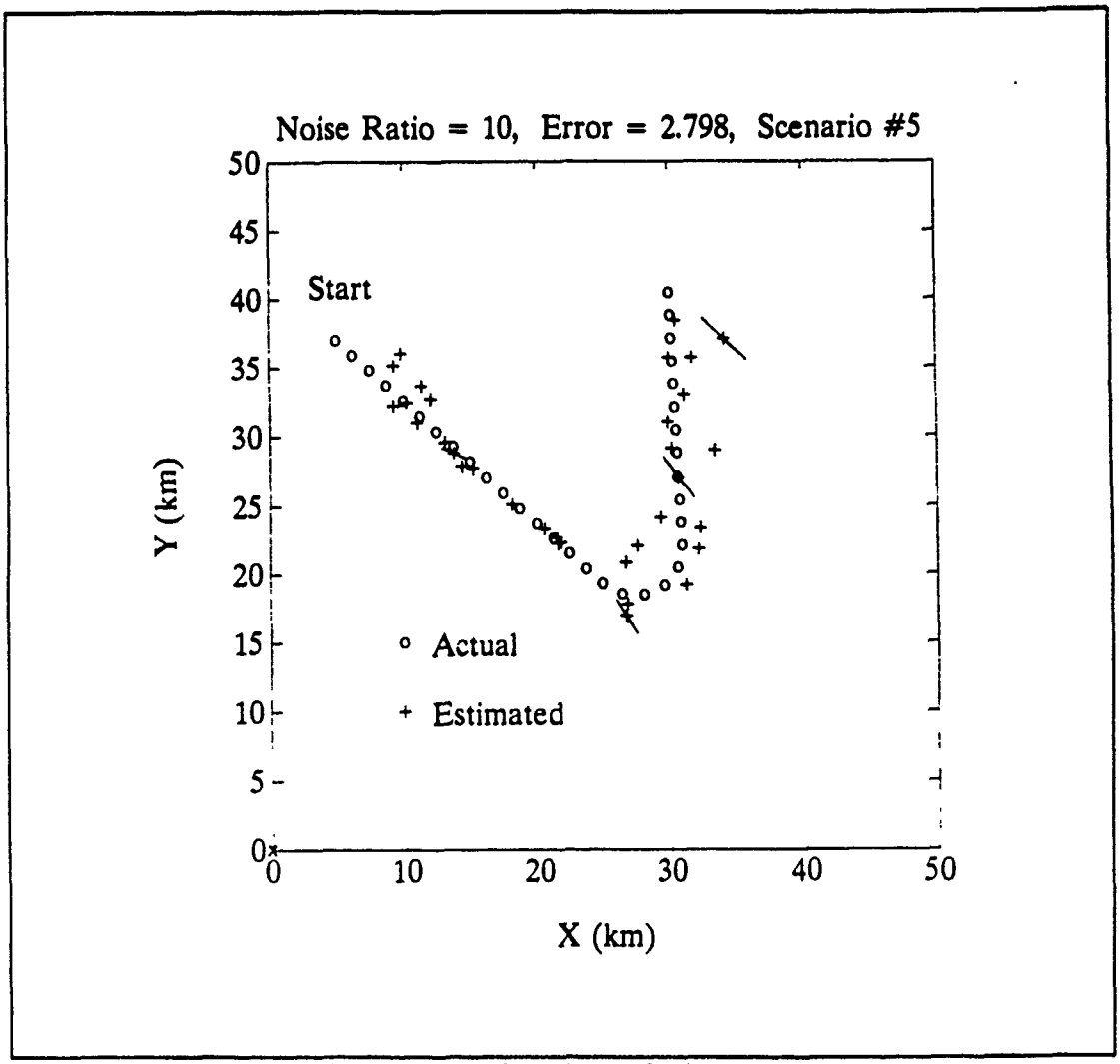

Figure 7.16: Target Track with Reset Covariance Gating

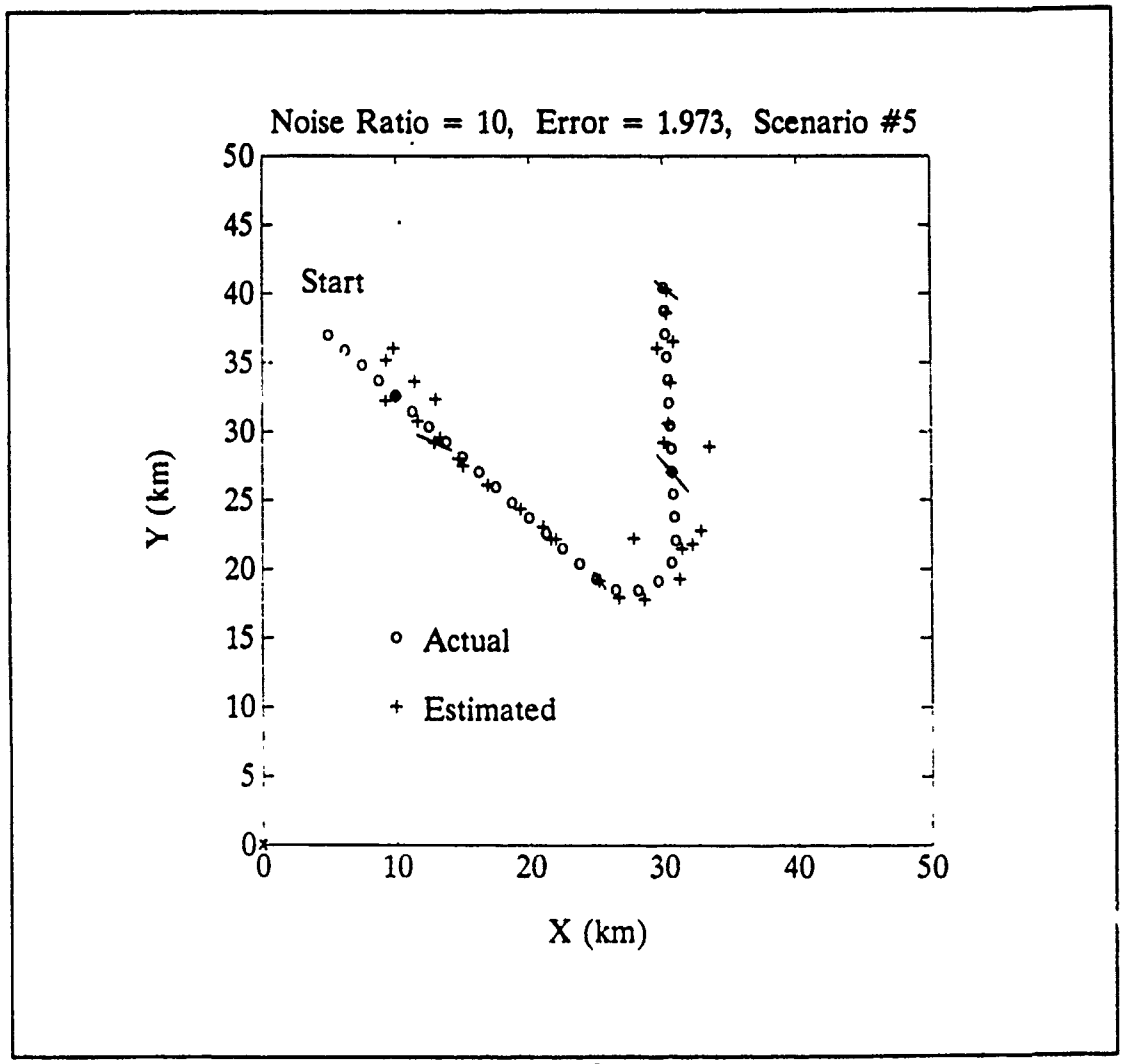

Figure 7.17: Target Track with Incremental, Correlated Gating

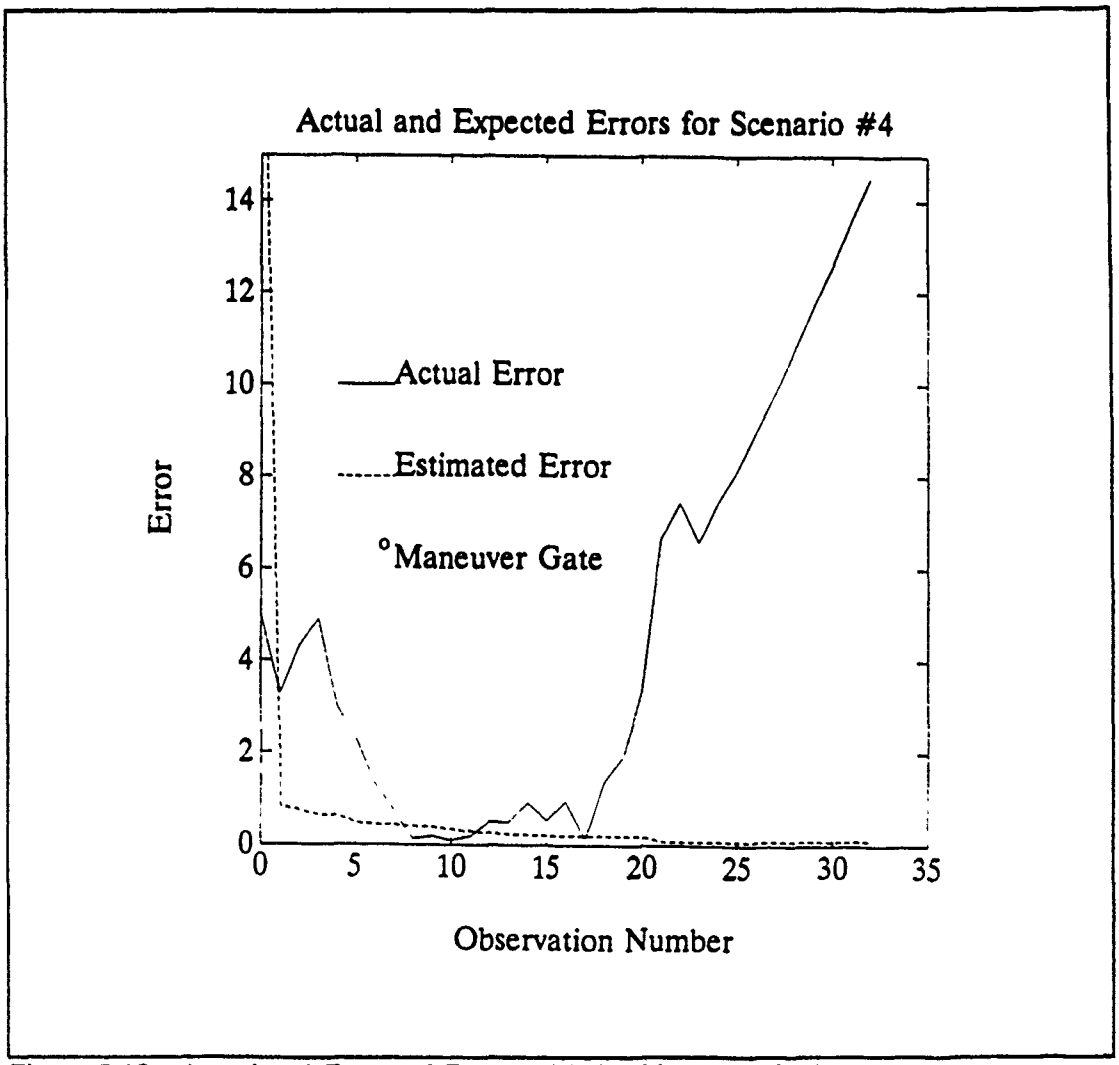

Figure 7.18: Actual and Expected Errors with No Maneuver Gating

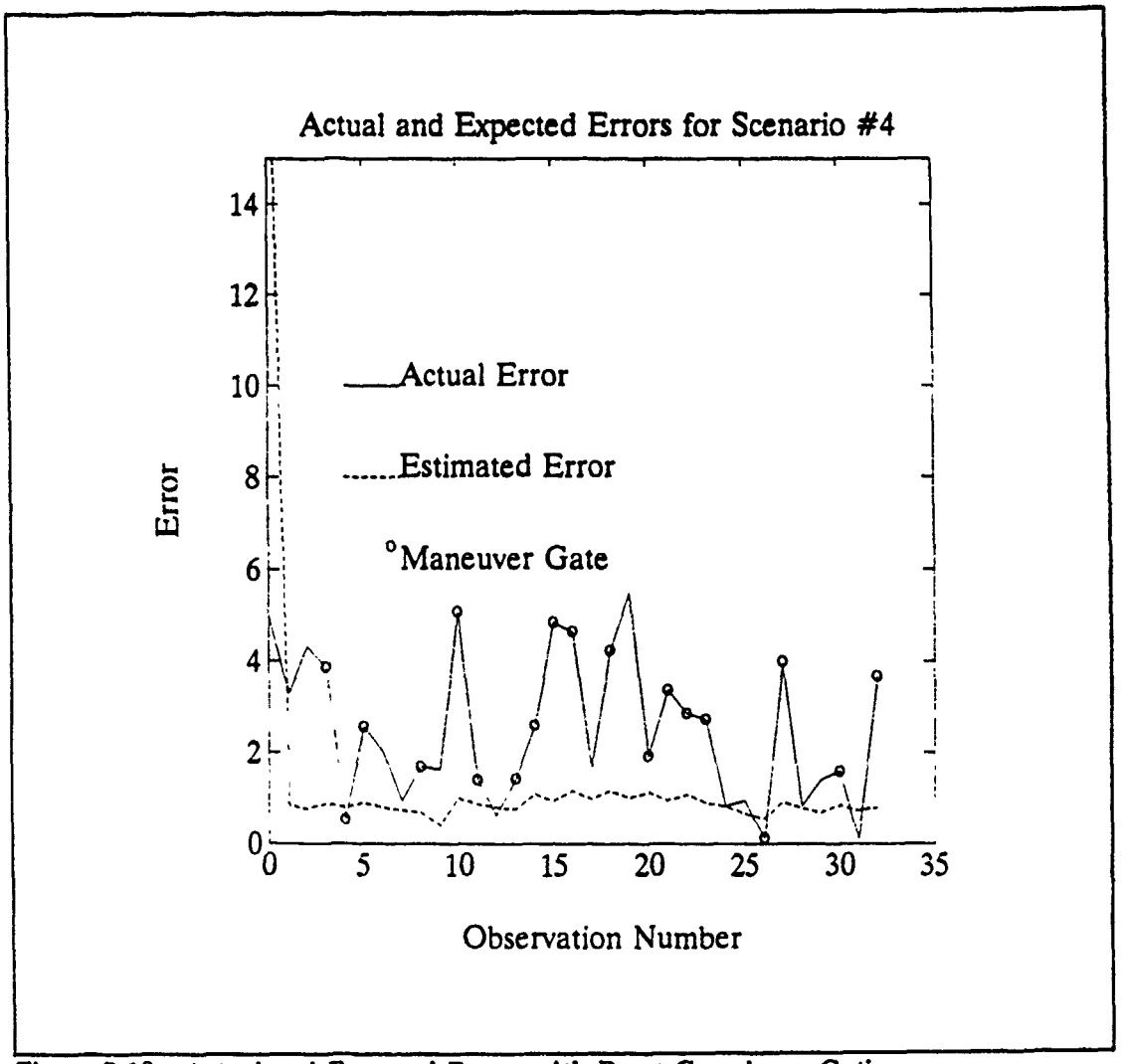

Figure 7.19: Actual and Expected Errors with Reset Covariance Gating

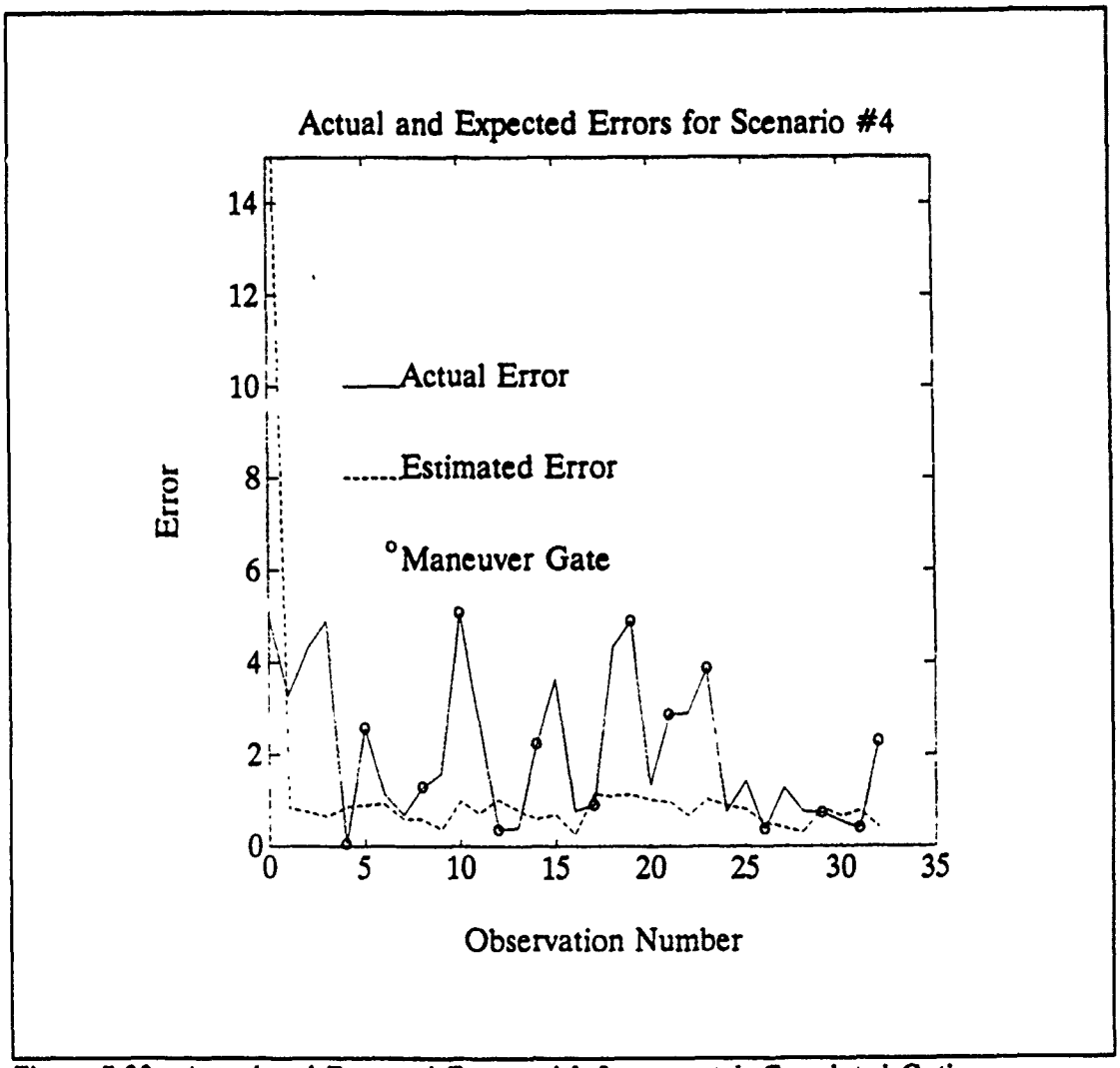

Figure 7.20: Actual and Expected Errors with Incremental, Correlated Gating

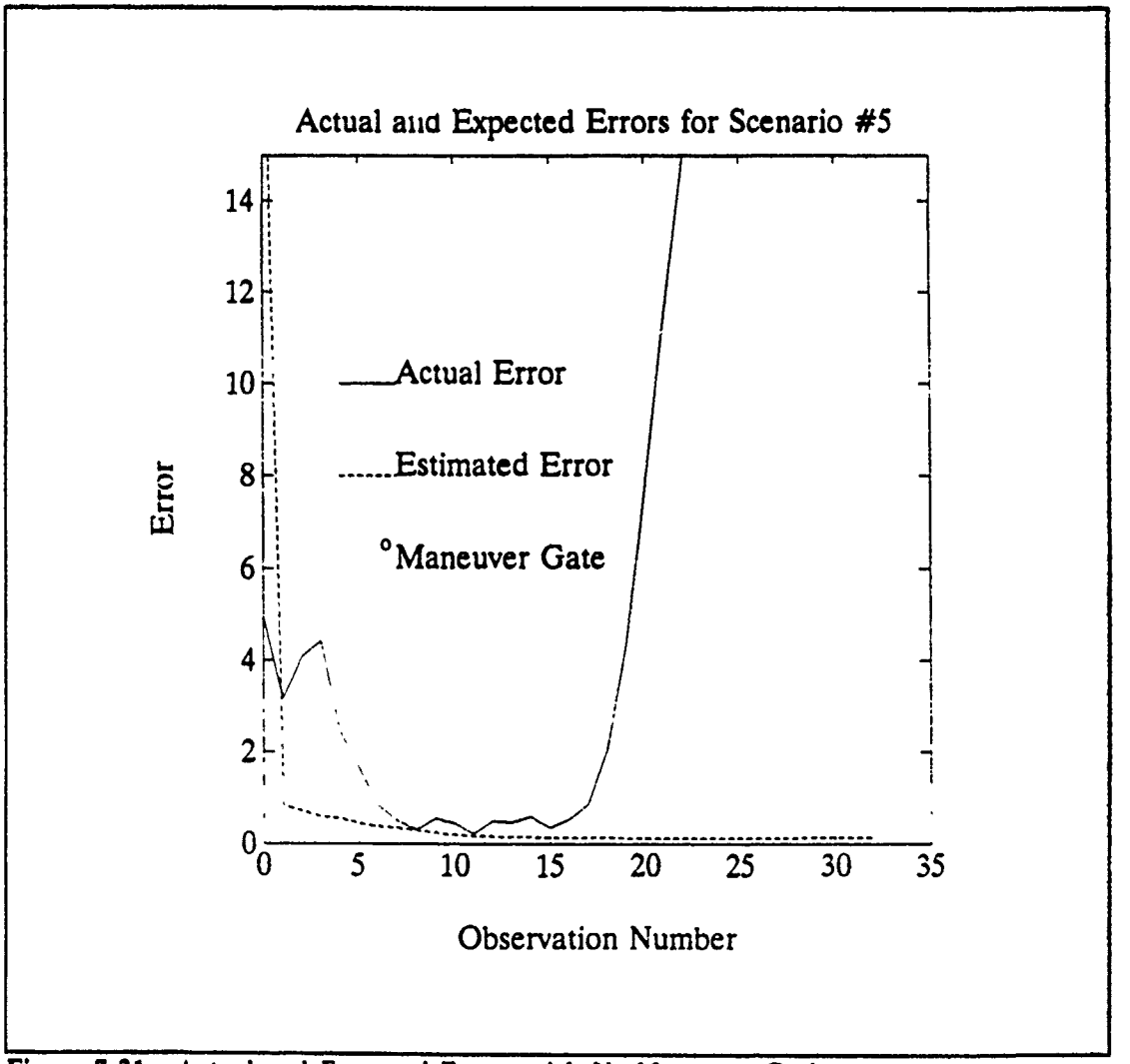

Figure 7.21: Actual and Expected Errors with No Maneuver Gating

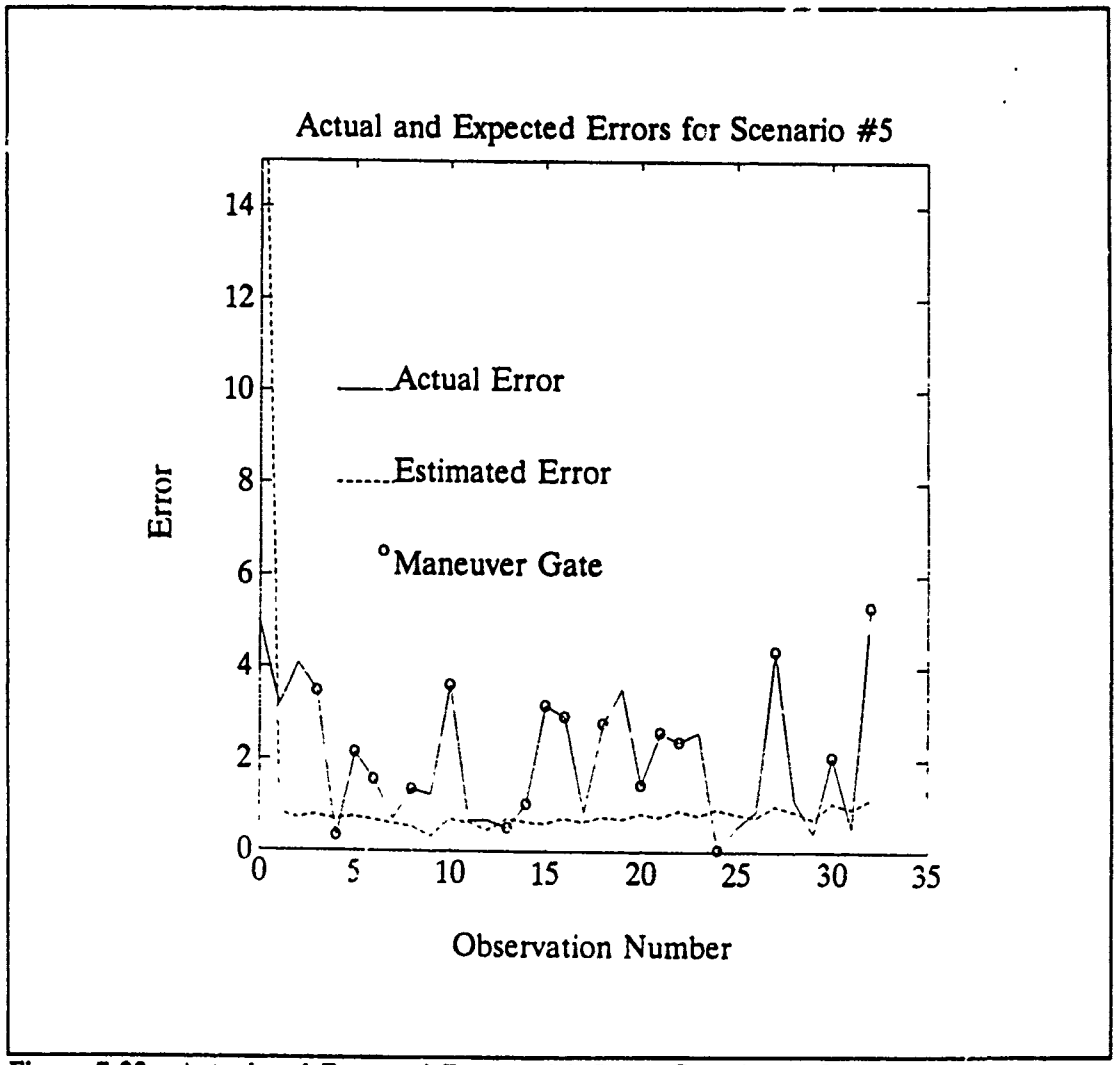

Figure 7.22: Actual and Expected Errors with Reset Covariance Gating

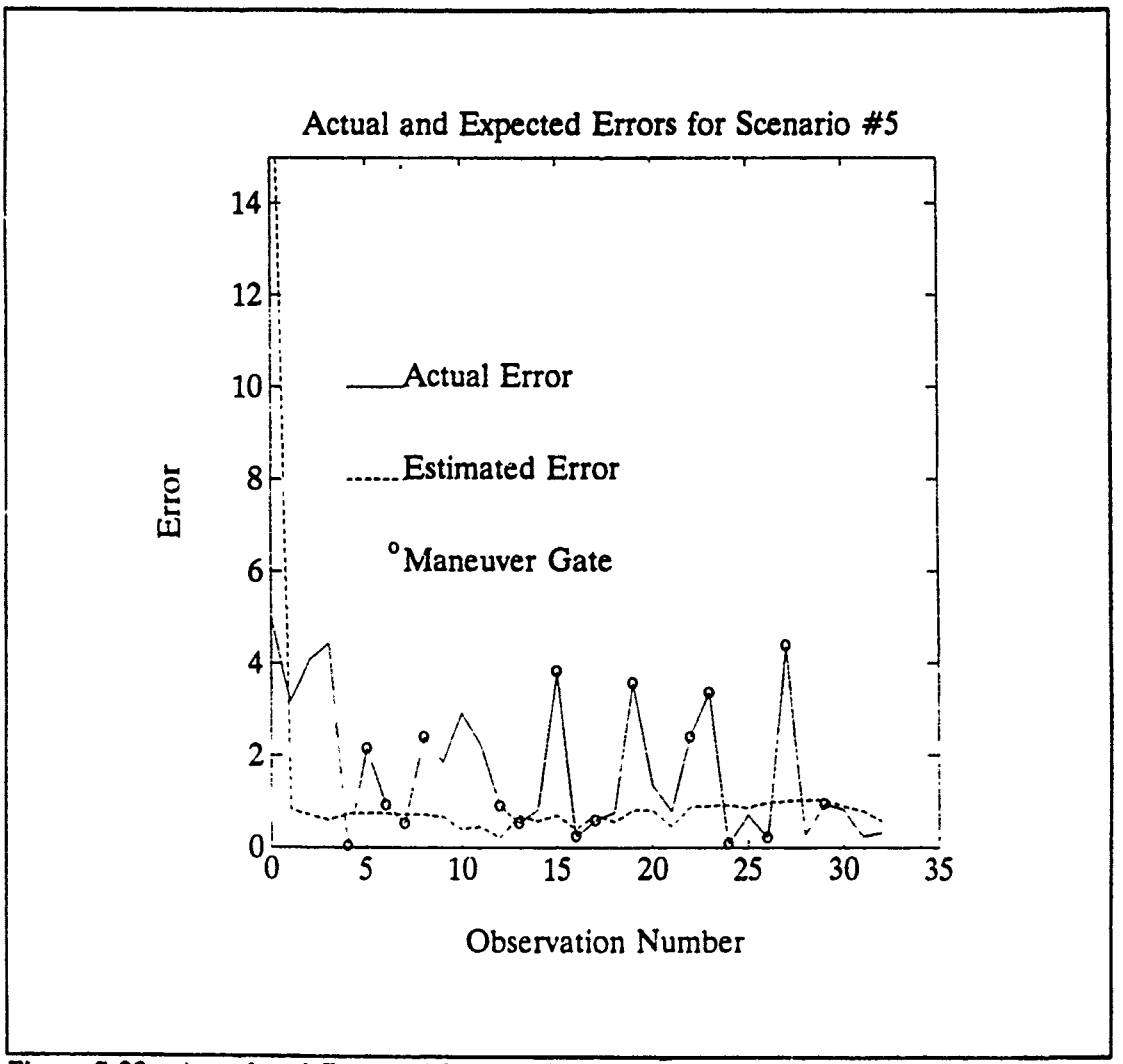

Figure 7.23: Actual and Expected Errors with Incremental, Correlated Gating

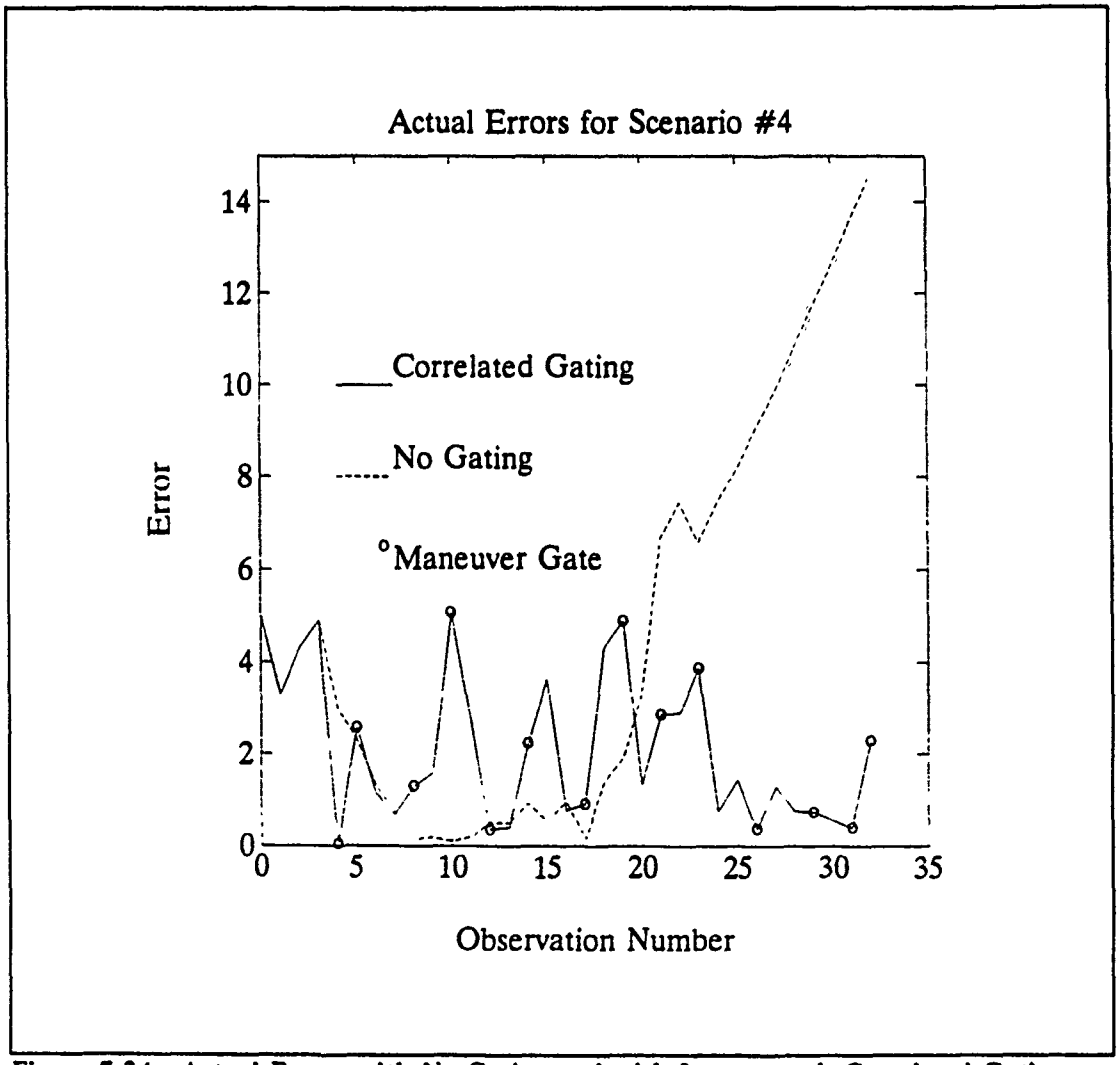

Figure 7.24: Actual Errors with No Gating and with Incremental, Correlated Gating

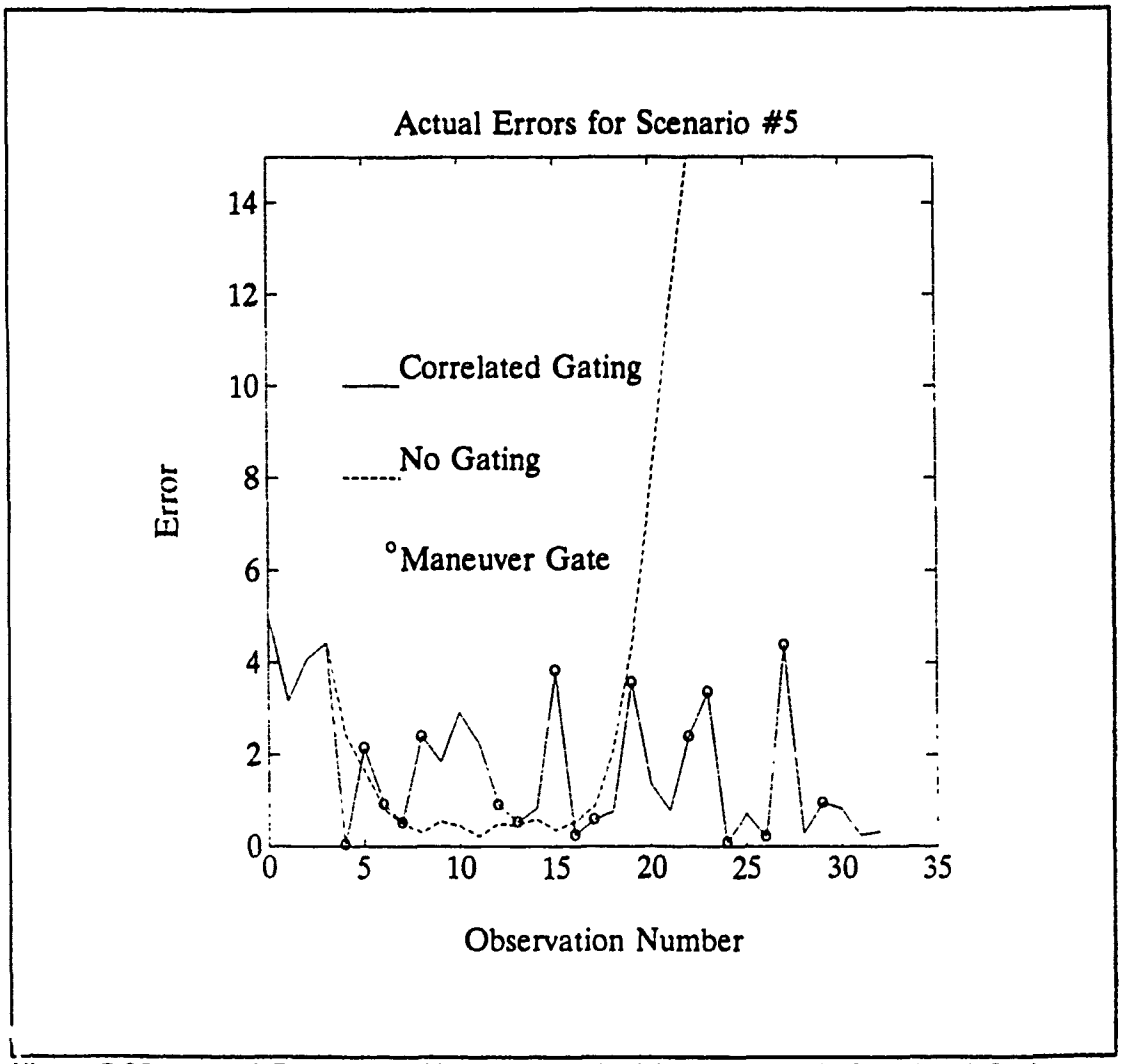

Figure 7.25: **Actual Errors with No Gating and with Incremental, Correlated Gating**

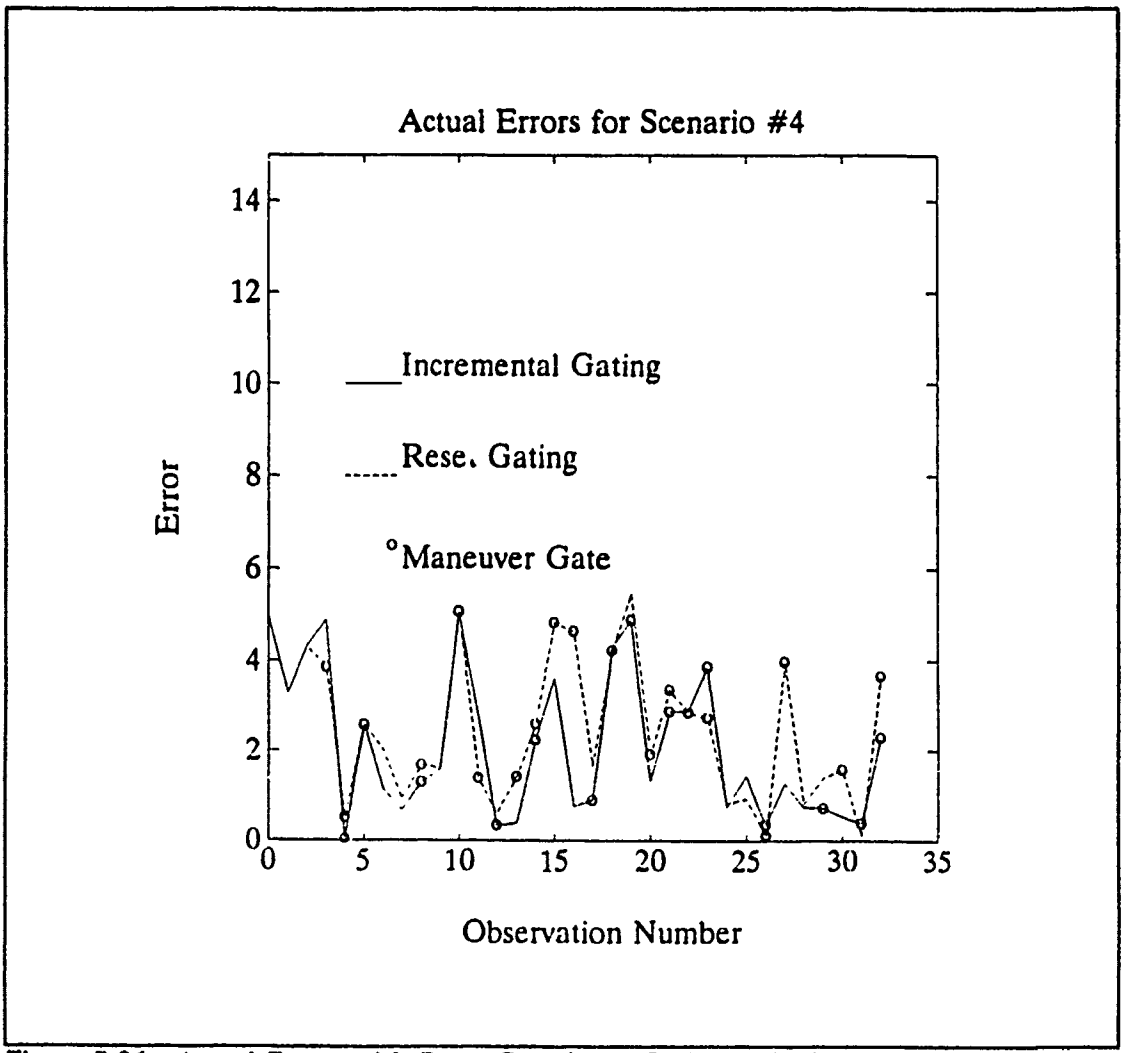

Figure 7.26: Actual Errors with Reset Covariance Gating and with Incremental, Correlated Gating

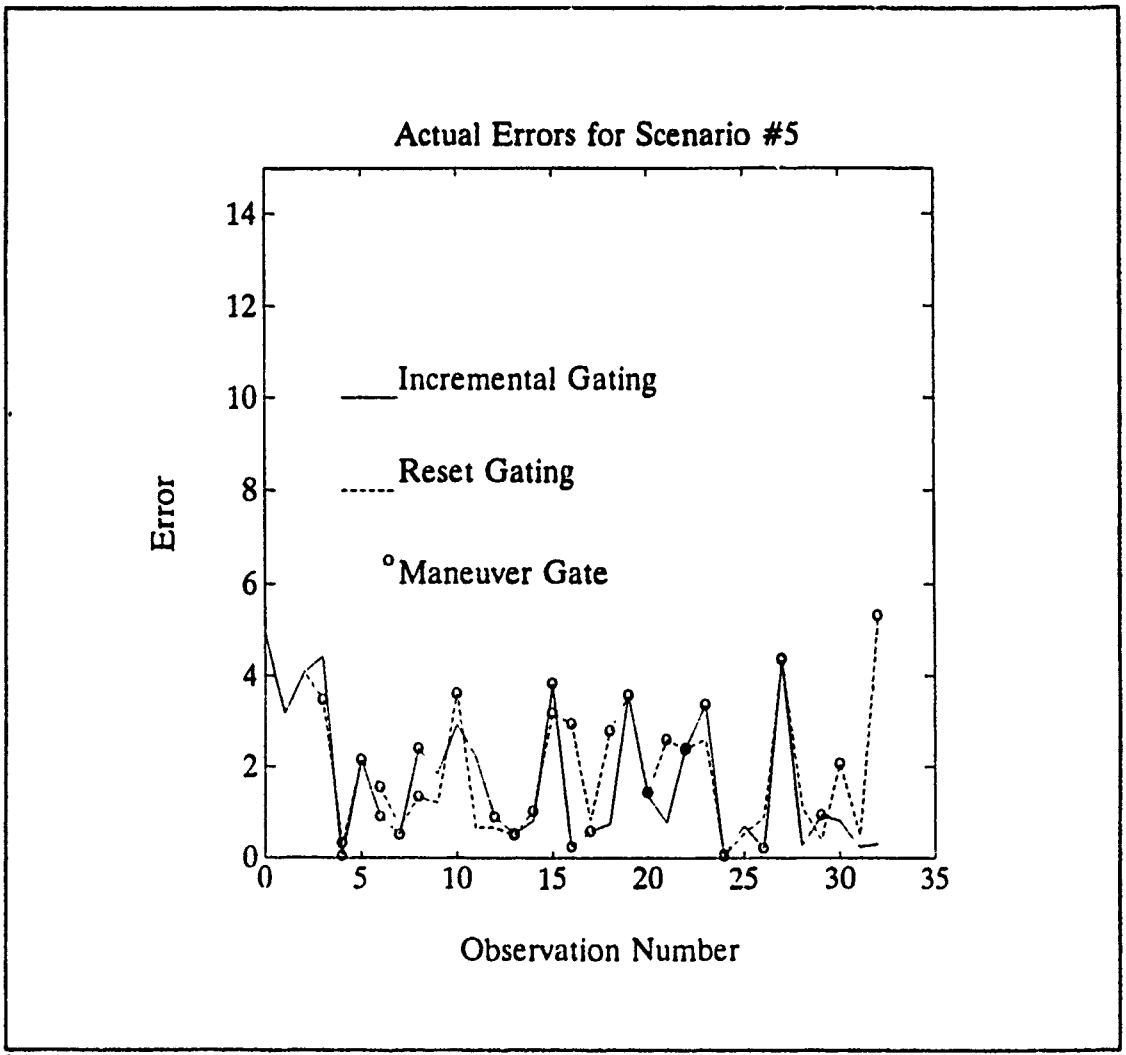

Figure 7.27: Actual Errors with Reset Covariance Gating and with Incremental, Correlated Gating

VIII. RESULTS

This chapter presents the results of the simulations run in Chapter VII. Following the structure of this report, the results are divided into two sections. The first section shows the results of the noise power estimation algorithm given in Chapter V, while the second section gives the results of the maneuver adaptation schemes described in Chapter VI.

A. NOISE ADAPTATION

The ability of the Kalman filter to estimate the noise power present in the observations is shown by Figures (7.1) and (7.2). Figure (7.1) shows that, for a non-maneuvering target, the Kalman filter produces a noise power estimate for the bearing noise which is a linear function of the actual noise power. This linear relationship is a weak function of the tracking geometry and holds for scenarios one through three. The linearity constant can be computed and used for further processing to provide even better estimates of both the bearing-noise power and the target state. This constant is likely a strong function of the state definition and should be found by simulation for each Kalman filter implementation.

This noise power estimation breaks down, however, when the range-noise power is estimated, as shown in Figure (7.2). Although scenario one (stationary target) provides the same linear relationship found in the bearing-noise estimate, the estimates for scenarios two and three provide little useful information for adapting the Kalman filter.

Using the noise power estimates, the Kalman filter's performance was tested and is shown in Figures (7.3) through (7.5). Each case shows that the filter's performance was improved by noise power adaptation as the observation noise was increased. This improvement came in spite of the poor estimates in the range channel.

B. MANEUVER GATING

The block model of the adaptive method used is shown in Figure (8.1). This figure shows that the adaptive algorithms use the residual and error covariance information from the Kalman filter and produce observation noise power estimates and maneuver gating information.

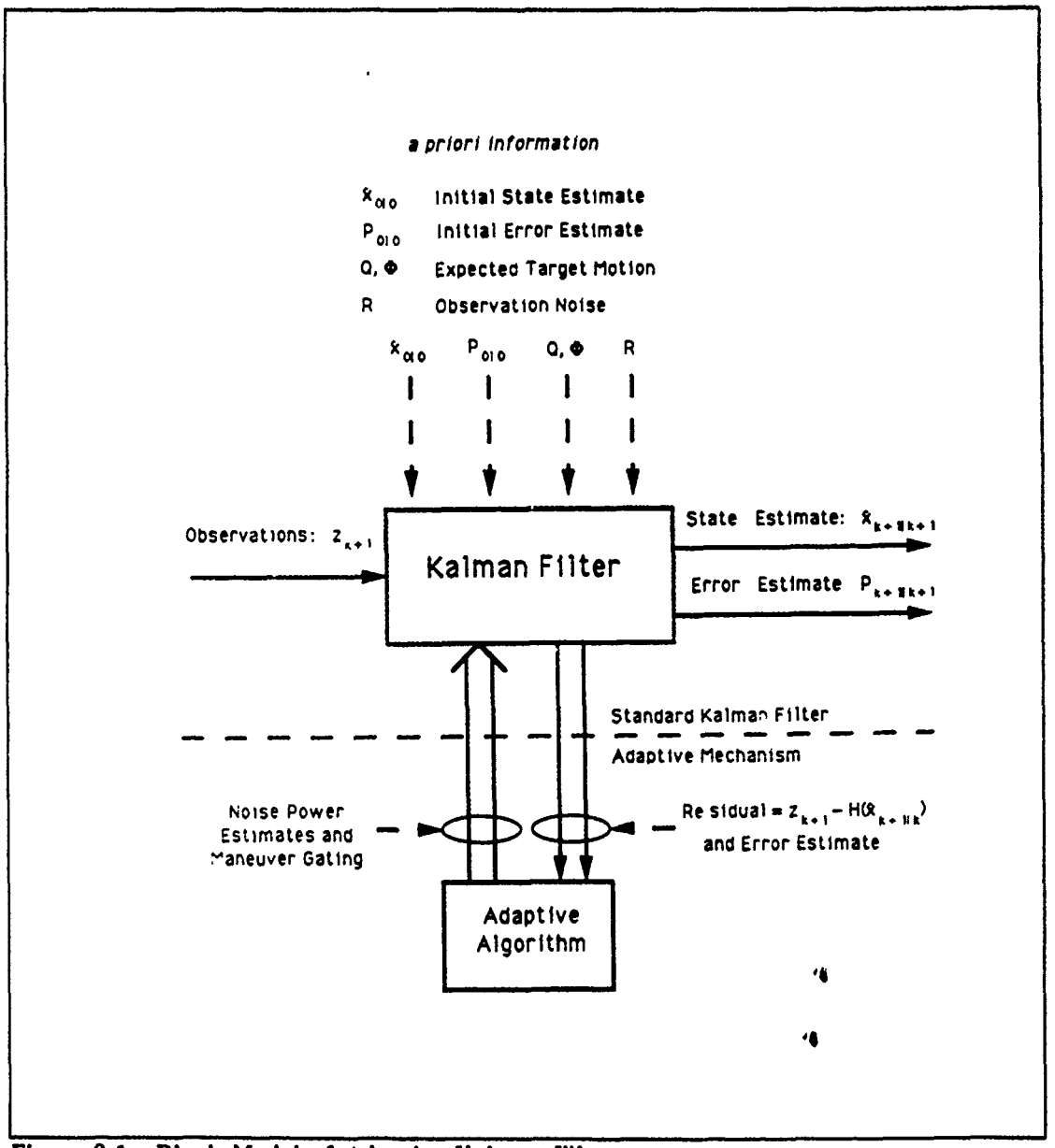

Figure 8.1: Block Model of Adaptive Kalman Filter

The first result found for maneuver gating is shown by Figures (7.6) and (7.7). These two graphs show that the performance of the Kalman filter in tracking a non-maneuvering target improves as the gate size increases. This improvement reaches a steady value at a gate size corresponding to a Mahalanobis Distance of about 2.25. But the filter's performance in tracking a maneuvering target improves as the gate size decreases.

These two results are in harmony with the way in which maneuver gating operates. When the target does not maneuver, any gating action will only incorrectly adjust the error covariance matrix resulting in an improperly large weighting to be given to the current observation. This is due to the gating algorithm incorrectly detecting anomalous noise as target maneuvering. When the target does maneuver, however, the gate size needs to be small so that the state estimates are able to remain close to the maneuvering target. The lower limit of this gate size is a value which is proportional to the noise power and inversely proportional to the magnitude of the target's acceleration.

Using the results given above, the gate size used in the simulations was set to an initial value of 2.25. This value was adapted down to a minimum of 0.2 whenever target maneuvering was detected, and reset to 2.25 when target maneuvers were complete. Adapting the gate size in this fashion provided experimentally optimal performance for this set of scenarios.

The rest of the simulations show that the overall best performance was obtained for tracking maneuvering targets by employing the technique of incremental, correlated maneuver gating. This method performed well for both maneuvering and non-maneuvering targets, while providing the greatest degree of noise rejection. Reset gating performed well for maneuvering targets, but poorly for non-maneuvering targets. This is because reset gating tends to "chase after" noisy observations so that its performance is

degraded in the face of anomalous noise. The non-adaptive Kalman filter suffered from divergence in the face of large noise power and a maneuvering target.

Figures (7.18) through (7.23) show that the method of incremental, correlated maneuver gating also provided the best estimate of its own error performance. The regular Kalman filter clearly gave unrealistic values for the estimated error, contributing to its own divergence.

IX. CONCLUSIONS AND RECOMMENDATIONS

We have shown that noise power estimation improves the performance of an extended Kalman filter in the face of anomalous observation noise. By estimating the noise power from the variance of the filter's residual we adapt the filter to compensate for varying noise power. Although the performance benefits were significant, much work needs to be done in this area to improve the noise power estimates further so the Kalman can provide still better state estimates.

We also introduced the method of correlated maneuver gating to adapt the Kalman filter to target dynamics. By [illegible]ly and temporally correlating the Mahalanobis Distance of the residual, the Kalman filter's performance is increased while tracking tangentially accelerating targets. These results should generalize to other applications of the extended Kalman filter whose state and observation spaces enjoy a one-to-one mapping.

Although we achieved our objectives of improving the performance of an extended Kalman filter in the face of significant observation noise and target maneuvering through adaptive techniques, there remains much fundamental work in this area for further improvements. Specifically, we feel that an approach incorporating the emerging potential of neural networks with the mathematical foundation of optimal estimation could provide new and exciting insight to an old problem.

APPENDIX A. TRANSFORMATION OF NOISE PROCESSES

The derivation of the extended Kalman filter equations assumed that the noise processes inherent in the physical system were zero-mean, additive, white, Gaussian noise (AWGN). This is normally a safe assumption, but the Kalman derivation also requires that the noise be Gaussian distributed in the state space. For the system as described in Chapter IV the noise processes are defined in spaces other than the state space.

This appendix will show that the transformation from the noise space to the state space will result in distributions which are Gaussian for the state excitation noise, and approximately Gaussian for the observation noise. Then the state excitation noise covariance matrix, Q, will be derived.

A. STATE EXCITATION NOISE

For the system used in this report the state is defined to be

$$\underline{X}_k = \begin{bmatrix} x_k \\ \dot{x}_k \\ y_k \\ \dot{y}_k \end{bmatrix} \tag{A.1}$$

where the dot denotes a time derivative and the underbar denotes a matrix (to avoid confusion with the state, x). This discrete-time state consists of x and y positions and velocities. The noise vector for this state is given as

$$w_k = \Gamma \begin{bmatrix} n_{xk} \\ n_{yk} \end{bmatrix} \tag{A.2}$$

where the state noise coefficient matrix is defined as

$$\Gamma = \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix} \tag{A.3}$$

and T is the sampling interval of the discrete system.

The problem is that the noise process is described in a coordinate system which is aligned with the direction of motion. As such, the noise elements describe the expected linear and tangential accelerations of the target. We will perform a linear transformation of the Gaussian noise into the state space. [Ref. 5]

The velocity of the target can be described in terms of its linear velocity and heading. This relationship, shown graphically in Figure (A.1), is given as

$$v_x = v_t \sin\Theta \tag{A.4}$$

and

$$v_y = v_t \cos\Theta. \tag{A.5}$$

Taking the time derivative of Equations (A.4) and (A.5) we get the accelerations in the x and y directions.

$$\begin{aligned} a_x &= \dot{v}_t \sin\Theta + v_t \dot{\Theta} \cos\Theta \\ &= \dot{v}_t \frac{v_x}{v_t} + v_t \dot{\Theta} \frac{v_y}{v_t} \\ &= \dot{v}_t \frac{v_x}{v_t} + \dot{\Theta} v_y \end{aligned} \tag{A.6}$$

$$\begin{aligned} a_y &= \dot{v}_t \cos\Theta - v_t \dot{\Theta} \sin\Theta \\ &= \dot{v}_t \frac{v_y}{v_t} - v_t \dot{\Theta} \frac{v_x}{v_t} \\ &= \dot{v}_t \frac{v_y}{v_t} - \dot{\Theta} v_x \end{aligned} \tag{A.7}$$

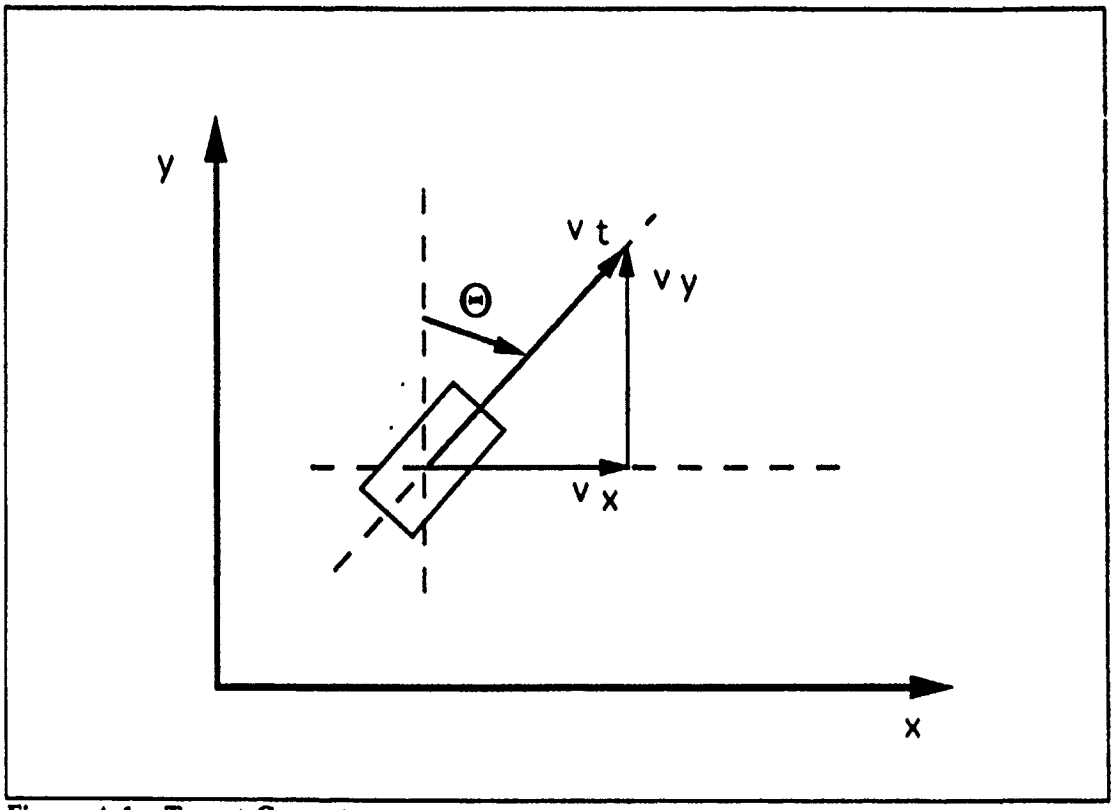

Figure A.1: Target Geometry

The noise is initially described by

$$E[\dot{v}_t] = E[\dot{\Theta}] = 0, \tag{A.8}$$

$$E[\dot{v}_t^2] = \sigma_v^2, \tag{A.9}$$

and

$$E[\dot{\Theta}^2] = \sigma_\Theta^2 \tag{A.10}$$

where the standard deviations in the linear and angular accelerations are given in Chapter IV. It is important to note that the random variables in Equations (A.8) through (A.10) are Gaussian distributed. Since a_x and a_y are linear combinations of these Gaussian variables (by Equations (A.6) and (A.7)), a_x and a_y are also Gaussian.

By squaring Equations (A.6) and (A.7) and taking the expectation of both sides, recalling Equation (A.8), we get expressions for the variances of a_x and a_y.

$$E[a_x^2] = \left(\frac{v_x}{v_t}\right)^2 \sigma_v^2 + v_y^2 \sigma_\theta^2 \qquad (A.11)$$

$$E[a_y^2] = \left(\frac{v_y}{v_t}\right)^2 \sigma_v^2 + v_x^2 \sigma_\theta^2 \qquad (A.12)$$

. We also find that the covariance of a_x and a_y is

$$E[a_x a_y] = E[a_y a_x] =$$

$$v_x v_y \left[\left(\frac{\sigma_v}{v_t}\right)^2 - \sigma_\theta^2 \right]. \qquad (A.13)$$

Armed with this, we can proceed to calculate the state excitation covariance matrix, Q. The matrix Q was defined to be

$$Q = E[w_k w_k^T]. \qquad (A.14)$$

Substituting Equation (A.2) into Equation (A.14), we get

$$Q = \Gamma \begin{bmatrix} E[a_x^2] & E[a_{xy}] \\ E[a_{xy}] & E[a_y^2] \end{bmatrix} \Gamma^T. \qquad (A.15)$$

By applying Equations (A.11) through (A.13) to Equation (A.15) we can calculate the covariance of the state excitation noise in the state space itself. In addition, Equations (A.6) and (A.7) ensure that the distributions are Guassian.

B. OBSERVATION NOISE

The coordinate transformation of the observation noise is non-linear due to the non-linear relationship between the observations and the state. The x and y positions of the target are related to the bearing and range of the observations by

$$x = \rho \sin\beta \tag{A.16}$$

and

$$y = \rho \cos\beta \tag{A.17}$$

where ρ and β are defined by Figure (A.2).

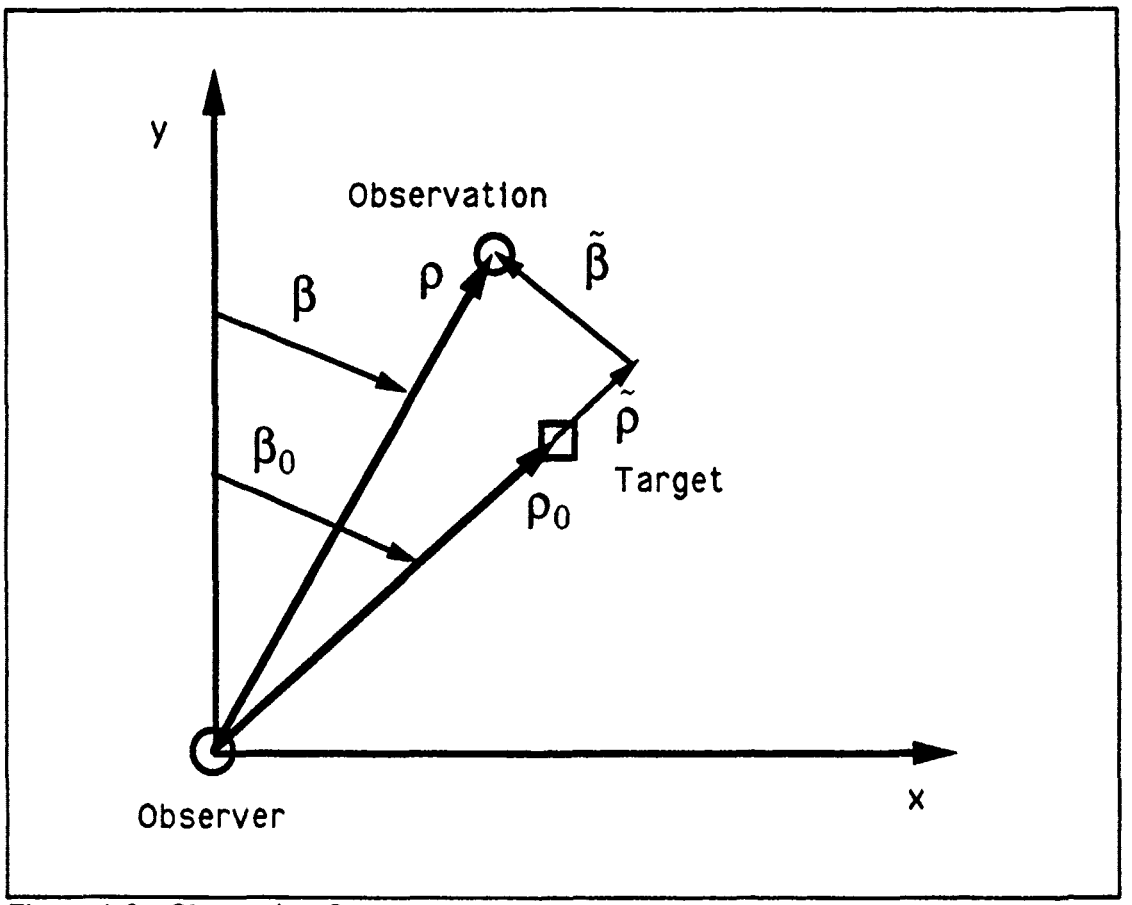

Figure A.2: Observation Geometry

By starting with the equation for the observations,

$$z_k = H(\underline{X}_k) + v_k, \tag{A.18}$$

we can describe the observations as

$$z_k = \begin{bmatrix} \beta_0 + \tilde{\beta} \\ \rho_0 + \tilde{\rho} \end{bmatrix} \tag{A.19}$$

where

$$H(\underline{X}_k) = \begin{bmatrix} \beta_0 \\ \rho_0 \end{bmatrix} \tag{A.20}$$

and

$$v_k = \begin{bmatrix} \tilde{\beta} \\ \tilde{\rho} \end{bmatrix}. \tag{A.21}$$

Putting Equations (A.20) and (A.21) into Equation (A.16) and expanding out the trigonometric functions [Ref. 6:p. 167] gives

$$\begin{aligned} x &= (\rho_0 + \tilde{\rho})\sin(\beta_0 + \tilde{\beta}) \\ &= (\rho_0 + \tilde{\rho})(\cos\beta_0\sin\tilde{\beta} + \sin\beta_0\cos\tilde{\beta}). \end{aligned} \tag{A.22}$$

For $|\tilde{\beta}| \ll 1$,

$$x = (\rho_0 + \tilde{\rho})(\tilde{\beta}\cos\beta_0 + \sin\beta_0), \tag{A.23}$$

and for $|\tilde{\rho}| \ll \rho_0$,

$$x = \rho_0\sin\beta_0 + \rho_0\tilde{\beta}\cos\beta_0 + \tilde{\rho}\sin\beta_0. \tag{A.24}$$

A similar treatment for y results in

$$y = \rho_0\cos\beta_0 - \rho_0\tilde{\beta}\sin\beta_0 + \tilde{\rho}\cos\beta_0. \tag{A.25}$$

Since x and y are shown by Equations (A.24) and (A.25) to be linear combinations of Gaussian random variables, then x and y are also Gaussian with means and variances of

$$M[x] = \rho_0 \sin\beta_0, \tag{A.26}$$

$$M[y] = \rho_0 \cos\beta_0, \tag{A.27}$$

$$V[x] = (\rho_0 \cos\beta)^2 \sigma_\beta^2 + \sin^2\beta_0 \sigma_\rho^2, \tag{A.28}$$

and

$$V[y] = (\rho_0 \sin\beta)^2 \sigma_\beta^2 + \cos^2\beta_0 \sigma_\rho^2 \tag{A.29}$$

where

$$\sigma_\rho^2 = E[\tilde{\rho}^2] \tag{A.30}$$

and

$$\sigma_\beta^2 = E[\tilde{\beta}^2] \tag{A.31}$$

This appendix has shown that the noise parameters of the system used in this report can be transformed into equivalent Gaussian vectors in the state space. This is not only necessary for the optimality of the linear Kalman filter, but it also gives physical meaning to the error ellipses derived in Appendix B.

APPENDIX B. ERROR ELLIPSES

Error ellipses provide a measure of confidence in the estimate of a parameter. Generally these ellipses indicate this confidence by visually displaying the smallest region which has a given probability of containing the actual value of the parameter. The size and shape of the ellipse is dependent on the distribution of the estimate, which is a random variable. This appendix develops the equations used to calculate and display the error ellipses for a two dimensional Gaussian error covariance matrix about a given estimate.

A. GAUSSIAN DISTRIBUTIONS

A Gaussian random variable, x, is any random variable whose probability distribution function is given by

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2\sigma^2}(x - \bar{x})^2} \tag{B.1}$$

where the mean and variance of x are given as

$$\bar{x} = E[x] \tag{B.2}$$

and

$$\sigma^2 = E[x^2]. \tag{B.3}$$

The expectation operator, E[], is defined in Equation (B.4).

$$E[g(x)] = \int_{-\infty}^{\infty} g(x)\ f(x)\ dx \tag{B.4}$$

When the random process is a vector random process, then the variable of interest is a vector of random variables. The Gaussian probability density function for a random vector, $\underline{X}$, is

$$f(\underline{X}) = \frac{1}{\sqrt{2\pi|K|}} e^{-\frac{1}{2}(\underline{X} - M)^T K^{-1} (\underline{X} - M)} \qquad (B.5)$$

where the mean and covariance of the vector $\underline{X}$ are given as

$$M = E[\underline{X}] \qquad (B.6)$$

and

$$K = E[(\underline{X} - M)(\underline{X} - M)^T]. \qquad (B.7)$$

The probability density function for a two-dimensional random vector is shown in Figure (B.1). This graph shows the probability curve to be a two-dimensional form of the standard Gaussian curve.

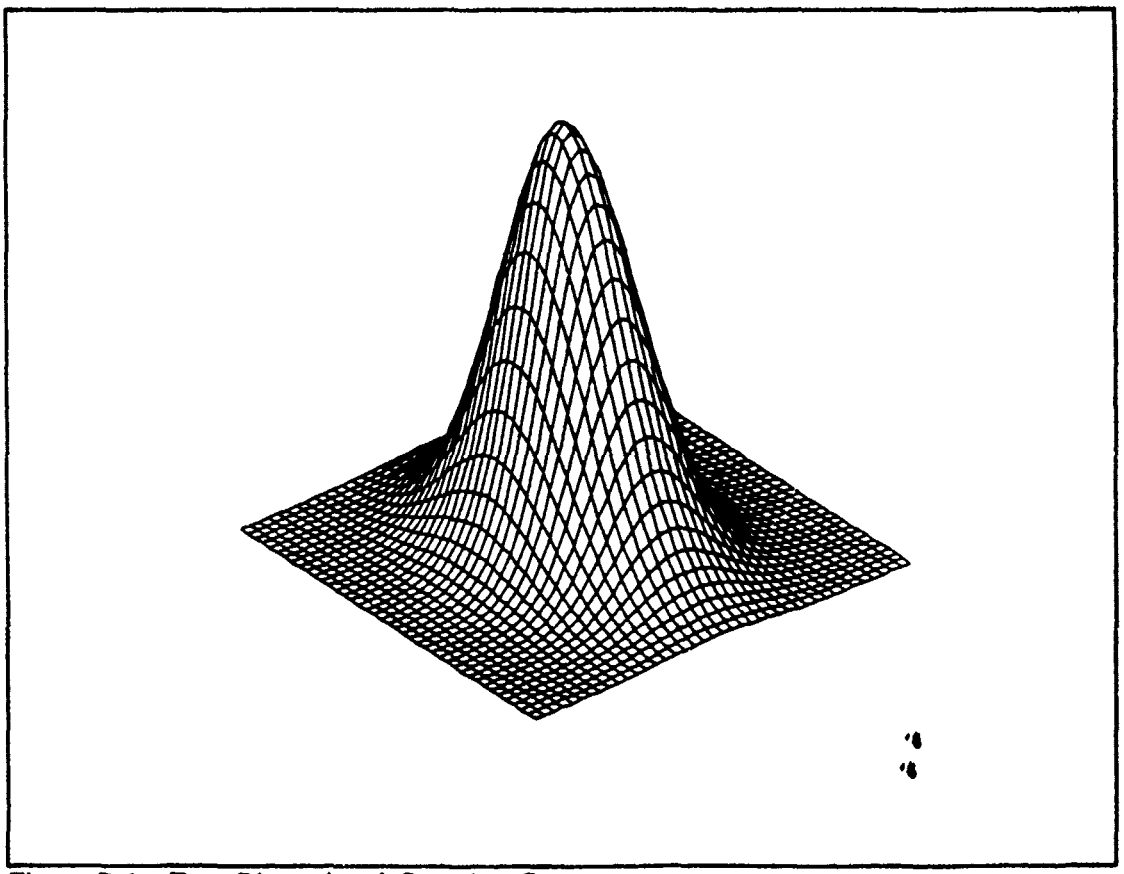

Figure B.1: Two-Dimensional Gaussian Curve

The contour plot of Figure (B.1) is shown in Figure (B.2) and plots contours of constant probability.

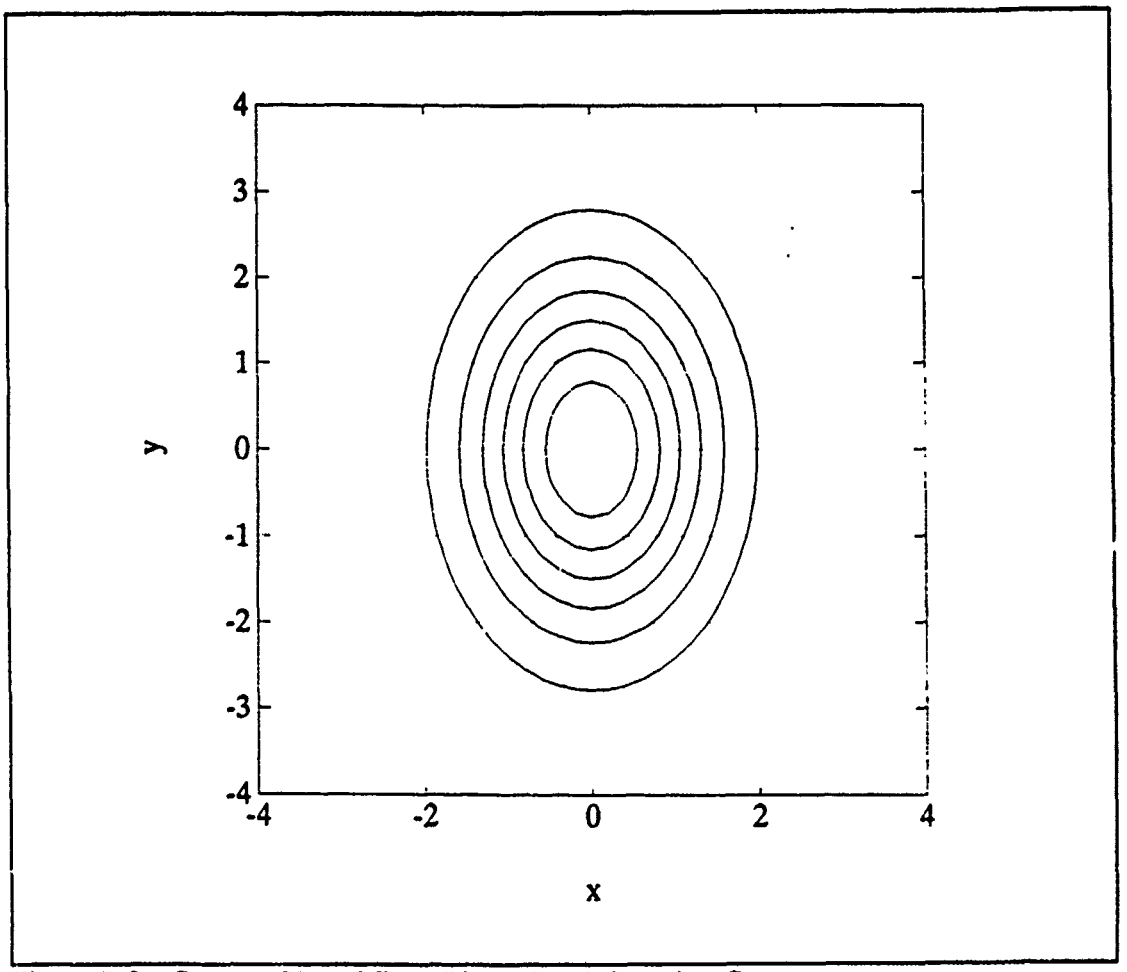

Figure B.2: Contour Plot of Two-Dimensional Gaussian Curve

The Gaussian distributions plotted in Figures (B.1) and (B.2) are for

$$M = \begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{B.8}$$

and

$$K = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}. \tag{B.9}$$

B. TWO DIMENSIONAL ERROR ELLIPSES

The most common error ellipse used to indicate confidence in a position estimate is a two dimensional representation of the position error covariance matrix. If the position state is higher than second order, then the state elements are taken two at a time to provide for a convenient graphical result. The following discussion is based upon a position estimate of the x and y coordinates of an unknown parameter.

In order to calculate the error ellipse, it is necessary to know the mean and covariance, M and K, of the parameter estimate. These matrices are defined by Equations (B.6) and (B.7) for the vector

$$\underline{X} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{B.10}$$

where the underscore is used to prevent confusion between the matrix $\underline{X}$ and the coordinate x.

In the previous section, the probability density function and probability contours were plotted for a two-dimensional Gaussian random vector. The contour plot is a locus of all points for which the probability is a constant. This satisfies the following equations:

$$f(\underline{X}) = \frac{1}{\sqrt{2\pi|K|}} e^{-\frac{1}{2}(\underline{X} - M)^T K^{-1}(\underline{X} - M)} = C_1$$

$$e^{-\frac{1}{2}(\underline{X} - M)^T K^{-1}(\underline{X} - M)} = C_2$$

$$(\underline{X} - M)^T K^{-1}(\underline{X} - M) = C \tag{B.11}$$

where the constant C will be determined shortly.

In order to separate the x and y coordinates in Equation (B.11) it is necessary that K be diagonal, i.e., that the off-diagonal terms of K be zero. This is not generally the case, however. We must transform the vector $\underline{X}$ into some vector $\underline{X}'$ for which the corresponding K' is diagonal. This is shown graphically in Figure (B.3), and is accomplished by means of eigenvector transformation.

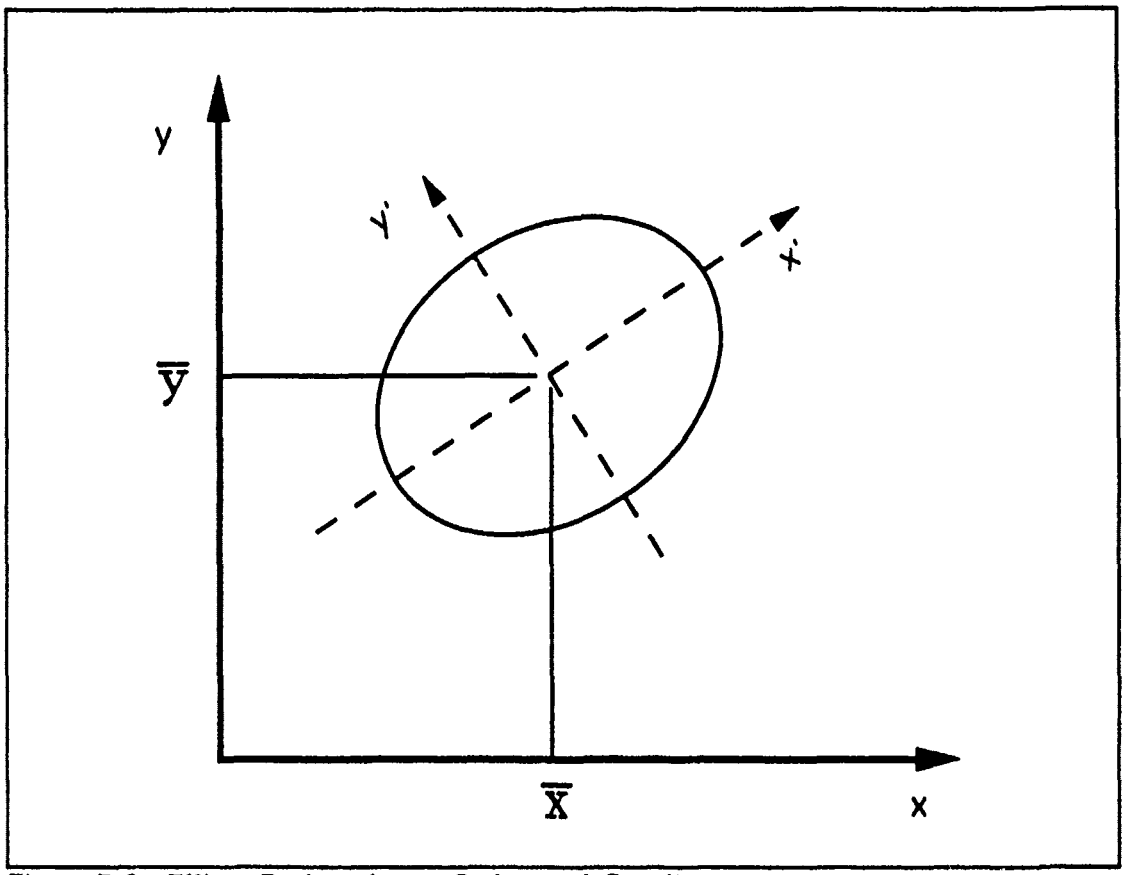

Figure B.3: Ellipse Projected onto Orthogonal Coordinates

1. Eigenvector Transformation

The eigenvalues, λ, and N x 1 column eigenvectors, v, of a square N x N matrix, K, satisfy

$$Kv = \lambda v. \tag{B.12}$$

This can be rearranged to

$$(K - \lambda I)v = 0. \tag{B.13}$$

There exists a non-zero v satisfying Equation (B.13) only if

$$|K - \lambda I| = 0. \tag{B.14}$$

When Equation (B.14) is expanded out it results in a polynomial in λ of the same degree as the size of the square matrix K. So, for an N x N matrix, K, there exist N (not necessarily unique) solutions to Equation (B.12). These solutions are the eigenvalues of K; the associated vectors, v, are the eigenvectors.

The eigenvectors are normally chosen so that they are orthonormal. They will then satisfy the following:

$$v_i^T v_j = \delta_{ij}. \tag{B.15}$$

When these orthonormal column vectors are combined into an N x N matrix, V,

$$V = [\ v_1 \mid v_2 \mid \cdots \mid v_N\ ] \tag{B.16}$$

they give the following useful result:

$$V^T = V^{-1}. \tag{B.17}$$

Combining Equations (B.12) and (B.16) we get the first stage of the transformation.

$$KV = K\,[\, v_1 \mid v_2 \mid \cdots \mid v_N \,]$$

$$= [\, Kv_1 \mid Kv_2 \mid \cdots \mid Kv_N \,]$$

$$= [\, \lambda_1 v_1 \mid \lambda_2 v_2 \mid \cdots \mid \lambda_N v_N \,]. \qquad \text{(B.18)}$$

Applying a little intuition we will now use Equations (B.15), (B.17), and (B.18) to arrive at the form of the transformation from K to K'.

$$V^T KV = V^T\,[\, \lambda_1 v_1 \mid \lambda_2 v_2 \mid \cdots \mid \lambda_N v_N \,]$$

$$= \begin{bmatrix} v_1^T \\ \cdots \\ v_N^T \end{bmatrix} [\, \lambda_1 v_1 \mid \lambda_2 v_2 \mid \cdots \mid \lambda_N v_N \,]$$

$$= \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \lambda_N \end{bmatrix} \qquad \text{(B.19)}$$

Defining K' to be the diagonal matrix of eigenvalues of K, we can write the transformation of K to K' in terms of the eigenvalues and eigenvectors.

$$K' = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \lambda_N \end{bmatrix} \qquad \text{(B.20)}$$

$$K' = V^T KV \qquad \text{(B.21)}$$

Putting Equation (B.7) into (B.21) we get the corresponding coordinate transformation,

$$K' = V^T E[(\underline{X} - M)(\underline{X} - M)^T] V$$

$$= E[V^T(\underline{X} - M)(\underline{X} - M)^T V]$$

$$= E[\underline{X}'\underline{X}'^T], \quad \text{(B.22)}$$

where

$$\underline{X}' = V^T(\underline{X} - M) \quad \text{(B.23)}$$

and

$$\underline{X} = V\underline{X}' + M. \quad \text{(B.24)}$$

Now that we have a transformed coordinate system with a diagonal covariance matrix we can proceed to the solution of the ellipse equations.

2. Ellipse Equations

Using the coordinate system defined by Equations (B.21) and (B.23), and using Equation (B.17), we can write Equation (B.11) as

$$(\underline{X} - M)^T K^{-1}(\underline{X} - M) = C$$

$$(\underline{X} - M)^T (VK'V^T)^{-1}(\underline{X} - M) = C$$

$$(\underline{X} - M)^T (VK'^{-1}V^T)(\underline{X} - M) = C$$

$$[V^T(\underline{X} - M)]^T K'^{-1}[V^T(\underline{X} - M)] = C$$

$$\underline{X}'^T K'^{-1}\underline{X}' = C = c^2 \quad \text{(B.25)}$$

where we introduce the new constant c^2 simply for later convenience.

Since K' is a diagonal matrix, we can expand Equation (B.25) into

$$\frac{x'^2}{\sigma_{x'}^2} + \frac{y'^2}{\sigma_{y'}^2} = c^2 \quad \text{(B.26)}$$

where

$$\sigma_{x'}^2 = \lambda_1 \quad \text{(B.27)}$$

and

$$\sigma_{y'}^2 = \lambda_2. \quad \text{(B.28)}$$

Equation (B.28) is the standard form for the equation of a two-dimensional ellipse. This can be solved parametrically by dividing through by c^2 and introducing a parametric variable, t.

$$\frac{x'^2}{c^2\sigma_{x'}^2} + \frac{y'^2}{c^2\sigma_{y'}^2} = 1$$
$$= \cos^2 t + \sin^2 t \qquad \text{(B.29)}$$

So our parameterized equations for x' and y' become

$$x' = c\sigma_{x'}\cos t \qquad \text{(B.30)}$$

and

$$y' = c\sigma_{y'}\sin t \qquad \text{(B.31)}$$

where $t \in [0,2\pi]$.

The constant c gives the size of the ellipse in terms of the two-dimensional standard deviation for the covariance matrix, K'. The probability of a point lying inside the ellipse is related to c by

$$Pr = \iint_S f(\underline{X})\ dxdy \qquad \text{(B.32)}$$

where S is the locus of points on the surface inside the ellipse defined by Equation (B.26) and the function $f(\underline{X})$ is defined by Equation (B.5). The relationship between c^2 and Pr for a two-dimensional Gaussian distribution is given on page 537 of Reference 6.

Solving Equation (B.26), substituting the solution into Equation (B.24), and plotting the points yields an error ellipse denoting the probability specified by the choice of the constant c^2. We have employed this method for the calculation and plotting of error ellipses in this report.

APPENDIX C. PROGRAM CODE

All of the simulations for this project were run on IBM-PC[1] class computers using the matrix manipulation language MATLAB[2], version 3.5f. This appendix contains the source code for all of the functions written in support of this project.

Only minor programming experience is required to understand these files. While MATLAB is similar to FORTRAN, MATLAB's control structures are much less complex. Comments are started by the percent sign (%) and continue to the end of the line.

To aid the reader in scanning and retyping these functions, each file is started on a new page. Although an analysis of the workings of these files is not necessary to understand this report, the curious (or skeptical) reader is highly encouraged to examine them closely.

The author neither claims nor desires to hold any copyright privileges on the source code. Written requests for the source code on computer disk should be sent either to the author, or to Professor Harold A. Titus. Address information can be found in the Initial Distribution List at the end of this report.

All of the files listed in the second section of this appendix provide general support for the main files listed in the first section. These support files are not specific to the simulations run for this report, but can be used for a variety of purposes.

---

1 IBM and IBM-PC are registered trademarks of IBM.

2 MATLAB is a registered trademark of The MathWorks, Inc..

A. INCLUDE.M

```
%  This file is a collection of constants which are used
%  by multiple files.
%  put the following line in the files which use this:
%
%  load include.mat

%  Stephen L. Spehn    30 Jan 1990

%  YES and NO are defined so as to index into YN_STR

NO              = 1;
YES             = 2;
YN_STR          = [ 'NO '
                    'YES' ];
LEFT            = 1;
RIGHT           = 2;

TIME_INTERVALS  = 32;
DELTA_T         = 10;          % 10 seconds

save include.mat;
```

B. DRIVER.M

```
% Stephen L. Spehn              31 Jan 1990

%  This file calls SENRUN01 with a variety of values for the
%  following parameters:
%
%  MD      Mahalanobis Distance
%  RAI     Residual Adaptation Interval
%  OCS     Observation Covariance Scale
%  ONV     Observation Noise Variance
%  SG      Spatial Gating
%  SN      Scenario Number
%  META_T  Make track graphs
%  META_R  Make rest of the graphs
%
% Number   x0      y0      vx0      vy0      g     Duration
%          (km)    (km)   (km/hr)  (km/hr)  (g's)   (sec)
%
%   1      5       37       0        0       0        0
%   2      5       37     450        0       0        0
%   3      5       37     400     -280       0        0
%   4      5       37     600       50     0.8       40
%   5      5       37     450     -300    -0.8       50
%
%  Target tracks for these scenarios are in the files
%  scen_1.mat through scen_5.mat.

!del sen01.*
!del sen02.*

%  Recompile included files
include,clear;
noise,clear;
scen_gen,clear;
```

```
%   DEFINED Constants
load include.mat;


MD  = [ .75 1 1.25 1.5 1.75 2 2.25 2.5 ];
OCS = [ 1 ];
ONV = [ 0.5 1 2 5 10 ];
RAI = [ TIME_INTERVALS ];
SG  = [ YES ];
SN  = [ 1 2 3 4 5 ];
META_T = NO;
META_R = NO;

for n1 = 1:length(SN)
   for n2 = 1:length(MD)
      senrun01(MD(n2),RAI,OCS,ONV,SG,SN(n1),META_T,META_R);
   end;
end;

!copy sen01.* sen02.*
!del sen01.*


MD  = [ 0 ];
OCS = [ 1 ];
ONV = [ 0.1 0.2 0.35 0.5 0.75 1 2 3.5 5 7.5 10 ];
RAI = [ 0 TIME_INTERVALS ];
SG  = [ NO ];
SN  = [ 1 2 3 4 5 ];
META_T = NO;
META_R = NO;

for n1 = 1:length(SN)
   for n2 = 1:length(RAI)
      senrun01(MD,RAI(RAI),OCS,ONV,SG,SN(n1),META_T,META_R);
   end;
end;

exit
```

C. SENRUN01.M

```
function    senrun01(MD,RAI,OCS,ONV,SG,SN,META_T,META_R)

% Stephen L. Spehn        31 Jan 1990

%    This is the driver file for the sensitivity test for the extended
% Kalman filter. The cumulative errors are shown as a function of noise
% parameters for the filter.
%
%    This file calls the following special functions:
%
%    BR_OBS
%    ERRELLIP
%    INT2STR2
%    KF_BR
%    KF_ERR
%    RESET
%    TIMESTR

% DEFINED Constants
load include.mat;

if nargin == 0
  MD  = 0;                          % Mahalanobis Distance
  RAI = TIME_INTERVALS;                  % Residual Adaptation Interval
  OCS = 1;                          % Observation Covariance Scale
  ONV = [ 0.2 0.5 1 2 3.5 5 7.5 10 ];   % Observation Noise Variance
  SG  = YES;                        % Spatial Gating
  SN  = 3;                          % Scenario Number
  META_T = YES;                        % Make meta for tracks
  META_R = NO;                         % Make rest of meta files
elseif nargin ~= 8
  error('Incorrect number of arguments to SENRUN01.');
end;
```

```
% House-cleaning
reset;


% Constants (Change to suit the problem.)
Erase_All_Files      = NO;
Make_Mat             = NO;
Make_Diary           = NO;
Make_Data            = YES;
Meta_Filename        = 'sen01.met';
Mat_Filename         = 'sen01.mat';
Diary_Filename       = 'sen01.dia';
Data_Filename        = 'sen01.dat';

Calculate_X0         = YES;
Iteration_Limit      = 3;       % Iterated Kalman
Iteration_Tolerance  = 2;       % km
Ellipse_Size         = 2;
Ellipse_Type         = 1;       % 0 Probability, 1 Mahalanobis Distance
Ellipse_Points       = 20;
Ellipse_Interval     = 8;

% Calculated Constants (Do not change these.)
Num_Sens_Runs  = length(ONV);



if Erase_All_Files == YES
   eval(['!del ',Meta_Filename]);
   eval(['!del ',Mat_Filename]);
   eval(['!del ',Diary_Filename]);
   eval(['!del ',Data_Filename]);
end;

if Make_Diary == YES
   eval(['diary ',Mat_Filename]);
end;
```

```
% System Information, in continuous time, for target, and initial guess
A = [ 0 1 0 0        % x
      0 0 0 0        % xdot
      0 0 0 1        % y
      0 0 0 0 ];     % ydot
BW = [ 0 0
       1 0
       0 0
       0 1 ];
x0  = [ 5            % Estimate of target initial state
        0
       37
        0 ];

% System disturbance and observation noise matrices.
% The observation noise matrix will be multiplied by the values
% in the vector Observation_Noise_Var.
W = [ 0.001   0            % linear acceleration
      0       0.001 ];     % angular acceleration
V = [ 0.0005 0.0           % bearing
      0.0    0.0005 ]*OCS; % range

% Initial Error Covariance Matrix
P0 = [ 200       0   0       0
         0  250000   0       0
         0       0 200       0
         0       0   0  250000 ];

% Read in the observation and target tracks, and the time vector
eval(['load scen_',int2str(SN),';']);

% Generate noiseless observation matrix.
obs = br_obs(xo,xt);

% Generate noisy observations.
load noise.mat;
for n1 = 1:Num_Noise_Runs;
   n1str = int2str(n1);
   eval(['temp_n = obs_noise',n1str,';']);
   temp_n(1,:) = temp_n(1,:) * sqrt(V(1,1));
```

```
  temp_n(2,:) = temp_n(2,:) * sqrt(V(2,2));
  eval(['obs_noise',n1str,' = temp_n;']);
end;

% Run the sensitivity analysis
IL = Iteration_Limit;
IT = Iteration_Tolerance;
IE = Calculate_X0;
total_err = zeros(Num_Sens_Runs,1);
res_vars  = zeros(Num_Sens_Runs,2);
res_means = zeros(Num_Sens_Runs,2);

% For each noise variance:
for n1 = 1:Num_Sens_Runs;
  n_var = ONV(n1);

  clc,
  fprintf('This computer is busy\n\n');
  fprintf('Scenario Number      = %1.0f\n',SN);
  fprintf('Mahalanobis Distance = %4.2f\n',MD);
  fprintf('Adaptation Interval  = %1.0f\n',RAI);
  fprintf('Covariance Scale     = %4.2f\n',OCS);
  fprintf(['Spatial Gating       = ',YN_STR(SG,:),'\n\n']);
  fprintf('Noise Power Ratio    = %4.2f\n',n_var);

  for n2 = 1:Num_Noise_Runs;
    eval(['obs_noise = obs_noise',int2str(n2),' * sqrt(n_var);']);
    n_obs = obs + obs_noise;

    % Get innovations, xy covariance, and estimates.
    [Z,GI,Res,Pxy,xe] = kf_br(xo,n_obs,x0,IE,...
                        A,BW,W,V,DTS/3600,MD,P0,RAI,IL,IT,SG);

    total_err(n1,1) = total_err(n1,1) + kf_err(xe,xt,Time/3600);
    res_vars(n1,:)  = res_vars(n1,:) + diag(cov(Res'))';
    res_means(n1,:) = res_means(n1,:) + mean(Res');
  end;
```

```
      eval(['GI',int2str(n1),' = GI;']);
      eval(['Res',int2str(n1),' = Res;']);
      eval(['Pxy',int2str(n1),' = Pxy;']);
      eval(['xe',int2str(n1),' = xe;']);

      % Plot track
      if META_T == YES
        n4 = 0;
        for n3 = 2:length(xe);
          if rem(n3,Ellipse_Interval) == 1
            n4 = n4+1;
            Pxy1 = Pxy(:,2*n3-1:2*n3);
            [xp,yp] = errellip([xe(1,n3);xe(3,n3)],Pxy1,Ellipse_Size,...
                               Ellipse_Points,Ellipse_Type);
            eex(:,n4) = xp;
            eey(:,n4) = yp;
          end;
        end;
        axis('square');
        axis([0 50 0 50]);
        plot(xt(1,:),xt(3,:),'o',xe(1,1:n3),xe(3,1:n3),'+',...
             xo(1,:),xo(3,:),'x',eex,eey,'-w');
        grid;
        title(['Noise Ratio = ',num2str(n_var),...
               ',  Error = ',num2str(total_err(n1,1)/Num_Noise_Runs),...
               ',  RAI = ',int2str(RAI),...
               ',  SG = ',YN_STR(SG,:)]);
        xlabel('X (km)'),ylabel('Y (km)');
        eval(['meta ',Meta_Filename]);
      end;

    end;
    res_vars  = res_vars / Num_Noise_Runs;
    res_means = res_means / Num_Noise_Runs;
    total_err = total_err / Num_Noise_Runs;
```

```
% Plot the results
if META_R == YES & Num_Sens_Runs > 1
  axis('normal');axis([1 2 3 4]);axis;
  semilogx(ONV,total_err);
  grid;
  xlabel('Normalized Observation Noise'),ylabel('Scaled Total Error');
  title(['Sensitivity of Extended Kalman Filter',...
        ', RAI = ',int2str(RAI),...
        ', SG = ',YN_STR(SG,:)]);
  eval(['meta ',Meta_Filename]);


  % Display the variance of the residual versus
  % the variance of the observation noise.
  axis('square');
  plot(ONV*V(1,1),Z(1,1));
  grid;
  xlabel('Observation Noise Variance'),ylabel('Variance of the Residual');
  title(['Residual Variance for Bearing',...
        ', RAI = ',int2str(RAI),...
        ', SG = ',YN_STR(SG,:)]);
  eval(['meta ',Meta_Filename]);

  plot(ONV*V(2,2),Z(2,2));
  grid;
  xlabel('Observation Noise Variance'),ylabel('Variance of the Residual');
  title(['Residual Variance for Range',...
        ', RAI = ',int2str(RAI),...
        ', SG = ',YN_STR(SG,:)]);
  eval(['meta ',Meta_Filename]);
end;


% Plot the total errors as a function of observation
if META_T == YES
  x_max = ceil((TIME_INTERVALS+1)/5)*5;
  axis('normal');axis([0 x_max 0 15]);axis;
  for n1 = Num_Sens_Runs:-1:1
    eval(['xe = xe',int2str(n1),';']);
    eval(['GI = GI',int2str(n1),';']);
```

```
      rms_err = sqrt( (xe(1,:) - xt(1,:)).^2 + (xe(3,:) - xt(3,:)).^2 );
      plot([0:length(xe)-1],rms_err,0,0);
      hold on;
      for n2 = 1:length(GI)
        if GI(n2) == YES
          plot(n2-1,rms_err(n2),'o');
        end;
      end;
    end;
    hold off;
    grid;
    xlabel('Observation Number'),ylabel('Error');
    title(['RMS Errors For All Runs',...
          ',  SG = ',YN_STR(SG,:)]);
    eval(['meta ',Meta_Filename]);
  end;

  % Save everything
  if Make_Mat == YES
    eval(['save ',Mat_Filename]);
  end;



  % Write the results to the data file.
  if Make_Data == YES
    if ~exist(Data_Filename)
      fprintf(Data_Filename,'Thesis Data for Stephen L. Spehn\n');
      fprintf(Data_Filename,['Time:  ',timestr,'\n\n']);
    else
      fprintf(Data_Filename,'\n\n');
    end;

    fprintf(Data_Filename,'Scenario Number      = %1.0f\n',SN);
    fprintf(Data_Filename,'Mahalanobis Distance = %4.2f\n',MD);
    fprintf(Data_Filename,'Adaptation Interval  = %1.0f\n',RAI);
    fprintf(Data_Filename,'Covariance Scale     = %4.2f\n',OCS);
    fprintf(Data_Filename,['Spatial Gating      = ',YN_STR(SG,:),'\n\n']);
    fprintf(Data_Filename,...
      'Noise Factor   Error     Residual Mean (B/R)    Residual Var (B/R)\n\n');
```

```
  for n1 = 1:Num_Sens_Runs;
    fprintf(Data_Filename,'  %6.2f       %6.2f',ONV(n1),total_err(n1,1));
    if res_means(n1,1) >= 0
      fprintf(Data_Filename,' ');
    end;
    fprintf(Data_Filename,'       %6.4f',res_means(n1,1));
    if res_means(n1,2) >= 0
      fprintf(Data_Filename,' ');
    end;
    fprintf(Data_Filename,'  %6.4f',res_means(n1,2));

    if res_vars(n1,1) >= 0
      fprintf(Data_Filename,' ');
    end;
    fprintf(Data_Filename,'       %6.4f',Z(1,1));
    if res_vars(n1,2) >= 0
      fprintf(Data_Filename,' ');
    end;
    fprintf(Data_Filename,'  %6.4f\n',Z(2,2));
  end;
end;

%  Cleanup
diary off;
```

D. KF_BR.M

```
function [Z,GI,Res,Pxy,xe] = kf_br(xobs,zobs,x0,IE,A,B,W,V,T,MD,P0,RAI,IL,IT,SG)
%
%KF_BR
%    [Z,GI,Res,Pxy,xe] = kf_br(xobs,zobs,x0,IE,A,B,W,V,T,MD,P0,RAI,IL,IT,SG)
% takes the following arguments:
%
% xobs     vector of observer positions
% zobs     matrix of noisy observations
% x0       initial estimate
% IE       Initial Estimate: 0 = use x0; 1 = use zobs(1)
% A        continuous time state dynamics matrix
% B        continuous time state excitation matrix
% W        state forcing function covariance matrix
% V        observation noise covariance matrix
% T        vector of observation intervals; these times are not running
%          times, but interval times
% MD       Mahalanobis Distance for maneuver gating: MD == 0 means no gating
% P0       initial error covariance matrix
% RAI      Residual Adapting Interval: RAI == 0 means no adaptation
%          The observation noise variance is compensated for the variance
%          of the residual at these intervals.
% IL       Iteration Limit
% IT       Iteration Tolerance
% SG       Spatial Gating: 1 for Yes, 0 for No
%
%    This function returns a row vector of gating information,
% an augmented column of innovations, an augmented row of error
% covariance matrices, and an augmented row matrix of estimated states.

% Stephen L. Spehn     30 Jan 1990

% Check arguments
if nargin ~= 15
  error('Incorrect number of arguments to KF_BR.');
end;
```

```
% DEFINED Constants
load include.mat;

% Allocate variables
kmax = length(T)+1;
Res = zeros(2,kmax-1);
Pxy = zeros(2,kmax*2);
Pxy(:,1:2) = [ P0(1,1) P0(1,3)
               P0(3,1) P0(3,3) ];
xe = zeros(length(x0),kmax);
if IE == NO
  xe(:,1) = x0;
else
  xe(1,1) = xobs(1,1) + zobs(2,1) * sin(zobs(1,1));
  xe(3,1) = xobs(3,1) + zobs(2,1) * cos(zobs(1,1));
end;
MDSide = NO * ones(1,kmax);     % Side of the residual (LEFT or RIGHT)
GI = NO * MDSide;               % Gating output
MDR = 0;                        % Mahalanobis Distance Ratio of the Residual

xi = zeros(length(x0),IL+1);
I  = eye(P0);
P  = P0;
KP = 1;
if SG == NO
  MD = 0;
end;


% Kalman Filter loop
next_adjust = RAI;
if RAI ~= 0
  Adapt = NO;
else
  Adapt = SG;
end;
oldt = T(1);
k = 0;
```

```
while 1
  k = k + 1;
  if k == kmax
    break;
  end;

  if k == 1
    P = P0;
  end;

  kbest = max(k,next_adjust-RAI);
  if (T(1) ~= oldt) | (k == 1)
    [phi,del] = c2d(A,B,T(1));
  end;
  oldt = T(k);

  % Time update of estimate
  xeminus = phi*xe(:,k);
  xi(:,1) = xeminus;

  % Do the iterated Kalman equations
  for k1 = 1:IL
    P1 = P;

    % Transform from polar W to rectangular Q
    varv  = W(1,1);
    varth = W(2,2);
    vt = sqrt((xi(2,k1))^2 + (xi(4,k1))^2);
    if vt ~= 0
      q11 = varv * (xi(2,k1) / vt)^2 + varth * (xi(4,k1))^2;
      q22 = varv * (xi(4,k1) / vt)^2 + varth * (xi(2,k1))^2;
      q12 = xi(2,k1) * xi(4,k1) * (varv / vt^2 - varth);
      q21 = q12;
      Q = [ q11 q12
            q21 q22 ];
    else
      Q = zeros(W);
    end;
    Q = del*Q*(del');
```

```
% Linearize H about best estimate
deltx = xi(1,k1);
delty = xi(3,k1);
r2 = deltx^2 + delty^2;
r = sqrt(r2);
H = [ delty/r2  0   -deltx/r2  0
      deltx/r   0    delty/r   0 ];

resid = br_resid(zobs(:,k+1),br_obs(xobs(:,k+1),xi(:,k1)));
MDSide(k+1) = br_side(xobs(:,k+1),xi(:,k1),zobs(:,k+1));

KP = 1;
MD1 == NO;
while 1
  P2 = KP*P1;
  P2 = phi*P2*phi' + Q;              % Projected error covariance
  vresid = H*P2*H' + V;              % Expected variance of the residual
  G = P2*H' / vresid;                % Kalman gain
  P2 = (I - G*H)*P2;                 % updated error covariance

  % Now check the Mahalanobis Distance of the residual
  if MD == 0
    MDR = 0;
  else
    MDR = (resid' / vresid * resid) / MD^2;
  end;
  if MDR > 1
    MD1 = YES;
  else
    MD1 = NO;
  end;
  if (MD == 0) | (MD1 == NO) | (Adapt == NO) ...
            | ((SG == YES)  &  (MDSide(k) ~= MDSide(k+1)))
    xi(:,k1+1) = xeminus + G*(resid - H*(xeminus - xi(:,k1)));
    break;
  else
    KP = 1.2*KP;
    GI(k+1) = YES;
  end;
end;
```

```
      % Check iteration tolerance
      if ( (xi(1,k1+1)-xi(1,k1))^2 + (xi(3,k1+1)-xi(3,k1))^2 ) < IT^2
        break;
      end;
    end;
    % Estimate is final iterated estimate
    xe(:,k+1) = xi(:,k1+1);
    P = P2;

    Res(:,k) = resid;
    Pxy(:,(2*k+1:2*k+2)) = [ P(1,1) P(1,3)
                             P(3,1) P(3,3) ];

    % Update the a priori observation noise covariance based on the residuals.
    % and start from the first observation.
    if k == next_adjust
      V = cov(Res');
      next_adjust = next_adjust + RAI;
      Adapt = YES;
      k = 0;
    end;

  end;
  Z = V;
```

E. BR_OBS.M

```
function    obs = br_obs(x_obs,x_tar)
%
%BR_OBS
%
%    This function takes a vector of observer positions
% and a vector of target positions and returns a matrix
% of observation vectors consisting of the bearing and range
% from the observer to the target.  The bearings are
% given in radians.
%
% x_obs    vector of observer positions,
% x_tar    vector of target positions.
%
% obs      matrix of bearing/range observation vectors

% Stephen L. Spehn     21 Nov 1989

% Check input arguments
if nargin ~= 2
  error('Incorrect number of arguments to BR_OBS.');
end;

% Get the x and y differences and calculate observations
x  = x_tar(1,:) - x_obs(1,:);
y  = x_tar(3,:) - x_obs(3,:);

b  = atan2(x,y);
r  = sqrt(x .^2 + y .^2);

obs = [ b
        r ];
```

F. BR_RESID.M

```
function    resid = br_resid(z_obs,z_est)
%
%BR_RESID
%
%    resid = br_resid(z_obs,z_est) takes the following arguments:
%
% z_obs    observation vector,
% z_est    estimated observation vector.
%
%    This function returns a vector of residuals for use in the
% Kalman filter estimate update equation.

% Stephen L. Spehn     18 Dec 1989

% Check number of input arguments
if nargin ~= 2
  error('Incorrect number of arguments to BR_RESID.');
end;

% Get difference
resid = z_obs - z_est;

% Normalize by putting angle in [-pi..pi]
resid(1,:) = resid(1,:) + 2*pi*(resid(1,:) < -pi);
resid(1,:) = resid(1,:) - 2*pi*(resid(1,:) >  pi);
```

G. BR_SIDE.M

```
function    side = br_side(xo,xe,z);
%
%side = br_side(xo,xe,z);
%
%    This function takes the observer's position, the current
% target estimate, and the bearing/range observation and returns
% an integer indicating on which side of the estimate the
% the observation lies.
%
% xo      observer location   [ x
%                             vx
%                             y
%                             vy ]
%
% xe      target estimate     [ x
%                             vx
%                             y
%                             vy ]
%
% z       observation         [ bearing
%                             range   ]

% Stephen L. Spehn     30 Jan 1990

% Check the input arguments
if nargin ~= 3
  error('Incorrect number of arguments to BR_SIDE.');
end;

% DEFINED Constants
load include.mat;

% Get the x and y difference between the estimate and the observation
xdiff = xo(1) + z(2) * sin(z(1)) - xe(1);
ydiff = xo(3) + z(2) * cos(z(1)) - xe(3);
```

```
% Rotate the velocity vector of the estimate by pi/2
xr = rotate_v([xe(2);xe(4)],pi/2);

% Get the component of the difference vector along this rotated vector
sign_r = xr(1) * xdiff + xr(2) * ydiff;

% If this is positive, then the observation lies to the right
if sign_r > 0
  side = RIGHT;
else
  side = LEFT;
end;
```

H. SCEN_GEN.M

```
function    scen_gen(n)

%   Stephen L. Spehn    31 Jan 1990

%   This file generates target tracks for the various scenarios
%   and stores them in mat files.
%
% Number   x0      y0      vx0      vy0      g    Duration
%          (km)    (km)   (km/hr)  (km/hr)  (g's)   (sec)
%
%    1      5       37       0        0      0        0
%    2      5       37      450       0      0        0
%    3      5       37      400     -280     0        0
%    4      5       37      600      50     0.8      40
%    5      5       37      450     -300    -0.8      50
%
%   Target tracks for these scenarios are in the files
%   scen_1.mat through scen_5.mat.

%   DEFINED Constants
load include.mat;


% Calculate time vector
DTS  = ones(1,TIME_INTERVALS)*DELTA_T;
Time = zeros(1,TIME_INTERVALS+1);
for n1 = 2:length(Time);
   Time(n1) = Time(n1-1) + DELTA_T;
end;

xo = zeros(4,TIME_INTERVALS+1);
xt = xo;
A = [ 0 1 0 0       % x
      0 0 0 0       % xdot
      0 0 0 1       % y
      0 0 0 0 ];    % ydot
```

```
%   scen_1.mat
xt0 = [ 5; 0; 37; 0 ];
xt = track_t(xt0,DTS/3600,A);
save scen_1.mat xo xt Time DTS;

%   scen_2.mat
xt0 = [ 5; 450; 37; 0 ];
xt = track_t(xt0,DTS/3600,A);
save scen_2.mat xo xt Time DTS;

%   scen_3.mat
xt0 = [ 5; 400; 37; -280 ];
xt = track_t(xt0,DTS/3600,A);
save scen_3.mat xo xt Time DTS;

%   scen_4.mat
g = 9.8 * 3600^2 / 1000;                    % g acceleration in km/hr^2
xt0 = [ 5; 600; 37; 50 ];
G4 = 0.8;
ACC_TIME = 4;
n  = length(DTS);
m1 = round(n/2);
m2 = n - m1 - ACC_TIME;
xt = track_t(xt0,DTS(1:m1-1)/3600,A);       % 1 .. m1

V = sqrt(xt(2,m1)^2 + xt(4,m1)^2);          % V in km/hr
theta = atan2(xt(2,m1),xt(4,m1));
ACC = G4*g;
da = ACC * DELTA_T / 3600 / V;              % angle change in radians
R = abs(V*V / ACC);                         % R in km
if ACC > 0                                  % right turn, Clyde
   xR = xt(1,m1) + R*cos(theta);
   yR = xt(3,m1) - R*sin(theta);
   alpha0 = 1.5*pi + theta;
else                                        % left tern
   xR = xt(1,m1) - R*cos(theta);
   yR = xt(3,m1) + R*sin(theta);
   alpha0 = 0.5*pi + theta;
   V = -V;
end;
```

```
for n1 = 1:ACC_TIME                          % 1 .. m1 + ACC_TIME
   alpha = alpha0 + da*n1;
   xt(1,m1+n1) = xR + R*sin(alpha);
   xt(3,m1+n1) = yR + R*cos(alpha);
   xt(2,m1+n1) =  V*cos(alpha);
   xt(4,m1+n1) = -V*sin(alpha);
end;
xt = [ xt track_t(xt(:,m1+ACC_TIME),DTS(m1+ACC_TIME+1:n)/3600,A) ];
save scen_4.mat xo xt Time DTS;

%   scen_5.mat
xt0 = [ 5; 450; 37; -400 ];
G5 = -0.8;
ACC_TIME = 5;
n  = length(DTS);
m1 = round(n/2);
m2 = n - m1 - ACC_TIME;
xt = track_t(xt0,DTS(1:m1-1)/3600,A);          % 1 .. m1
V = sqrt(xt(2,m1)^2 + xt(4,m1)^2);             % V in km/hr
theta = atan2(xt(2,m1),xt(4,m1));
ACC = G5*g;
da = ACC * DELTA_T / 3600 / V;                 % angle change in radians
R = abs(V*V / ACC);                            % R in km
if ACC > 0                                     % right turn, Clyde
   xR = xt(1,m1) + R*cos(theta);
   yR = xt(3,m1) - R*sin(theta);
   alpha0 = 1.5*pi + theta;
else                                           % left tern
   xR = xt(1,m1) - R*cos(theta);
   yR = xt(3,m1) + R*sin(theta);
   alpha0 = 0.5*pi + theta;
   V = -V;
end;
for n1 = 1:ACC_TIME                          % 1 .. m1 + ACC_TIME
   alpha = alpha0 + da*n1;
   xt(1,m1+n1) = xR + R*sin(alpha);
   xt(3,m1+n1) = yR + R*cos(alpha);
   xt(2,m1+n1) =  V*cos(alpha);
   xt(4,m1+n1) = -V*sin(alpha);
end;
```

```
xt = [ xt track_t(xt(:,m1+ACC_TIME),DTS(m1+ACC_TIME+1:n)/3600,A) ];
save scen_5.mat xo xt Time DTS;
```

I. TRACK_T.M

```
function    x = track_t(x0,tvec,A)
%
%TRACK_T
%
%    x = track_t(x0,tvec,A) takes the following arguments:
%
% x0    initial target position
% tvec  vector of observation intervals; these times are not running
%       times, but interval times.  i.e. tvec = [1 1.5 .2 .7 .9 .5 1]
% A     continuous time state transition matrix
%
%    This function returns a matrix of the target states.

% Stephen L. Spehn     15 Nov 1989

% Check arguments
if nargin ~= 3
  error('Incorrect number of arguments to TRACK_T.');
end;

% Check for consistency in the size of the inputs
[rx,cx] = size(x0);
if cx ~= 1
  error('x0 must be n x 1.');
end;
[rA,cA] = size(A);
if rA ~= cA
  error('A must be square.');
end;
if rx ~= cA
  error('x0 and A must be conformable.');
end;

% Allocate and initialize x and B
N = length(tvec);
x = zeros(rx,N+1);
```

```
x(:,1) = x0;
B = zeros(x0);

% Loop through and calculate new positions
oldt = tvec(1);
for k = 1:N

  % Only recalculate phi and del when necessary
  if (tvec(k) ~= oldt) | (k == 1)
    [phi,del] = c2d(A,B,tvec(k));
  end;

  % Discrete state projection
  x(:,k+1) = phi*x(:,k);

  % Save the current time interval for the check above
  oldt = tvec(k);
end;
```

J. KF_ERR.M

```
function    err = kf_err(xe,xt,T)
%
%KF_ERR
%
%     err = kf_err(xe,xt,T) takes the following arguments:
%
%  xe     vector of estimated target positions,
%  xt     vector of actual target positions,
%  T      vector of Times of estimates.
%
%     This function returns the total weighted error:
%
%        err = sum(abs(distance) * Time)

%  Stephen L. Spehn     24 Nov 1989

%  Check arguments
if nargin ~= 3
   error('Incorrect number of arguments to KF_ERR.');
end;

%  Compute the error
dif_ = sqrt((xt(1,:) - xe(1,:)).^2 + (xt(3,:) - xe(3,:)).^2);
err  = sum( dif_ .* T );
```

K. NOISE.M

```
% This file generates the suite of noise matrices for SENRUN01.M

% Stephen L. Spehn        31 Jan 1990

% defined constants
load include.mat

Num_Noise_Runs      = 30;      % the more runs, the better the average
Standard_Noise_Seed = 1989;    % 1 .. 2^31 - 1
Use_Standard_Noise  = YES;     % YES: same random numbers every day

% Generate a set of observation noise matrices.
rand('normal');
if Use_Standard_Noise == YES
  rand('seed',Standard_Noise_Seed);
else
  rand('seed',sum(clock)*100);
end;
for n1 = 1:Num_Noise_Runs;
  eval(['obs_noise',int2str(n1),' = rand(2,TIME_INTERVALS+1);']);
end;

save noise.mat
```

L. SUPPORT FILES

1. Errellip.m

```
function    [x,y] = errellip(M,K,Pr,n,MD)
%
%errellip(M,K,Pr,n,MD)
%
%    This function is used to generate vectors of x and y
% coordinates for the plotting of error ellipses.  It takes
% the following arguments:
%
%       M     Mean Vector (2 x 1)
%       K     Covariance Matrix (2 x 2)
%       Pr    Probability (0 .. 1)
%       n     Number of points to compute
%       MD    Compute by Mahalanobis Distance vice probability
%             0 = Probability
%             1 = Mahalanobis Distance
%
% and returns x and y vectors of the confidence ellipse for the
% given probability or Mahalanobis Distance.

% Stephen L. Spehn      15 Nov 1989

% Check the input arguments
if nargin ~= 5
  error('Incorrect number of arguments to ERRELLIP.');
end;

% Compute the Mahalanobis Distance for the ellipse
if MD == 1
  c = abs(Pr);
else
  Pr = max(min(.995,Pr),.005);
```

```
  % Cubic spline fit for the ellipse constant.
  % Using this method is a compromise between speed and accuracy.
  pp = [ 12;
       .005; .01; .025; .05; .1; .25; .5; .75; .9; .95; .975; .99; .995;
       4.0;
       4.190809197869072e+1;  4.190809197872625e+1;  -2.118721291999464e+1;
       3.970929262330003;     2.15930349191602;      5.955793735647319e-1;
       1.628822411111034e+1;  8.24375856514019e+1;   2.725551521454689e+3;
       4.176985755223079e+2;  2.087990965023806e+5;   2.087990965023842e+5;
      -3.810356328008524e-1;  2.4758574o8793011e-1;  2.133449885921905;
       5.444089169224142e-1;  1.140048306271903;      2.111734877634124;
       2.55841940780766;      1.477458749113522e+1;  5.187150103426589e+1;
       4.6070422925247e+2;    4.920316224166436e+2;  9.887990965023757e+3;
       2.020857475864537;     2.02019022643493;      2.055905760926949;
       2.122852230998054;     2.20707509215777;      2.694842569743673;
       3.862381141104123;     8.195632865839841;     1.819254614465004e+1;
       4.382133265898673e+1;  6.763972895071458e+1;  2.23340067762321e+2;
    .01; .0201; .0506; .103; .211; .575; 1.39; 2.77; 4.61; 5.99; 7.38; 9.21 ];

  c = ppval(pp,Pr);
end;

% Get Eigenvectors and Eigenvalues of the Covariance matrix
[Evec,Eval] = eig(K);
sigx = sqrt(Eval(1,1));
sigy = sqrt(Eval(2,2));

% Parameterize the ellipse equations by the angle, t
t = 0:(2*pi/n):(2*pi);
xv = sigx*c*cos(t);
yv = sigy*c*sin(t);

% Translate back to the original coordinates
% and add in the mean (center of the ellipse)
x = (xv*Evec(1,1) + yv*Evec(1,2) + M(1))';
y = (xv*Evec(2,1) + yv*Evec(2,2) + M(2))';
```

2. Int2str2.m

```
function    str = int2str2(i)
%
%INT2STR2
%
%    str = int2str2(i) returns a two character string with the
% value of i as a zero padded integer to two places.  Examples
% are given below.
%
%    i = 5       str = '05'
%    i = 50      str = '50'
%    i = 500     str = '500'
%    i = -5      str = '-5'

% Stephen L. Spehn     30 Nov 1989

str = int2str(i);

if (i < 10) & (i >= 0)
   str = ['0',str];
end;
```

3. Reset.m

```
function    reset
%
%RESET
%
%     This function is used to reset various MATLAB parameters
% so that the current simulation will be unaffected by previous
% runs.

% Stephen L. Spehn      25 Nov 1989

hold off;
subplot(111);
axis('normal');axis([1 2 3 4]);axis;
clg,clc;
clear;
```

4. Rotate_v.m

```
function    new_vector = rotate_v(vector,angle)
%
%new_vector = rotate_v(vector,angle);
%
%    This function takes a vector of x and y coordinates and an
% angle of rotation specified in radians.  It returns the
% rotated vector.
%
% vector        [ x
%                 y ];
% angle         angle of rotation in radians (positive CW)
%
% new_vector    [ x'
%                 y' ];

% Stephen L. Spehn     17 Jan 1990

% Check input arguments for number and size
if nargin ~= 2
  error('Incorrect number of arguments to ROTATE_V.');
end;

[rv,cv] = size(vector);
if (rv ~= 2) | (cv ~= 1)
  error('Input vector incorrect size.');
end;

% Calculate transformation matrix and rotate the vector
cosl = cos(angle);
sinl = sin(angle);

T = [  cosl  sinl
     -sinl  cosl ];

new_vector = T * vector;
```

5. Timestr.m

```
function    tstr = timestr
%
%TIMESTR
%
%     This function returns the time and date in a string, i.e.,
%
%          1525, 28 Nov 1989

%  Stephen L. Spehn      28 Nov 1989

months = [' Jan '
          ' Feb '
          ' Mar '
          ' Apr '
          ' May '
          ' Jun '
          ' Jul '
          ' Aug '
          ' Sep '
          ' Oct '
          ' Nov '
          ' Dec ' ];

td = fix(clock);
tstr = [int2str2(td(4)),int2str2(td(5)),', ',...
        int2str(td(3)),months(td(2),:),int2str(td(1))];
```

LIST OF REFERENCES

1. R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Trans. ASME, J. Basic Eng.*, Vol. 82D, No. 1, pp. 35-46, March 1960.

2. R. E. Kalman and R. S. Bucy, "New Results in Linear Filtering and Prediction Theory," *Trans. ASME, J. Basic Eng.*, Vol. 83D, No. 1, pp. 95-108, March 1961.

3. Analytical Sciences Corporation, Technical Staff, *Applied Optimal Estimation*, Arthur Gelb, ed., The M.I.T Press, Cambridge, Massachusetts, 1974.

4. Gene F. Franklin and J. David Powell, *Digital Control of Dynamic Systems*, pp. 308-309, Addison-Wesley, Reading, Massachusetts, 1980.

5. William J. Galinis, "Fixed Interval Smoothing Algorithm for an Extended Kalman Filter for Over-the-Horizon Ship Tracking," Master's thesis, Naval Postgraduate School, Monterey, California, 1989.

6. *CRC Standard Mathematical Tables*, William H. Beyer, ed., CRC Press, Inc., West Palm Beach, Florida, 1978.

BIBLIOGRAPHY

T. E. Bullock and Suwanchai Sangsuk-Iam, "Optimal evasive maneuver detection," Final Report on Contract FO8635-83-K-0191, The Air Force Armament Laboratory, Eglin Air Force Base, Florida, AFATL-TR-84-02, January 1984.

C. K. Chui and G. Chen, *Kalman Filtering with Real-Time Applications*, Springer-Verlag, New York, 1987.

George R. Cooper and Clare D. McGillem, *Probabilistic Methods of Signal and Systems Analysis*, Holt, Reinhart and Winston, New York, 1986.

Dick P. Dee, Stephen E. Cohn, Amnon Dalcher, and Michael Ghil, "An efficient algorithm for estimating noise covariances in distributed systems," *IEEE Trans. on Automatic Control*, Vol. AC-30, No. 11, pp. 1057-1062, 1985.

Bernard Friedland, *Control System Design*, McGraw-Hill, Inc., New York, 1986.

Graham C. Goodwin and Kwai Sang Sin, *Adaptive Filtering Prediction and Control*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1984.

Stanley I. Grossman, *Multivariable Calculus, Linear Algebra, and Differential Equations*, Harcourt Brace Jovanovich, Inc., San Diego, California, 1986.

Donald E. Kirk, *Optimal Control Theory*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1970.

Raman K. Mehra, "Optimal input signals for parameter estimation in dynamic systems - survey and new results," *IEEE Trans. on Automatic Control*, Vol. AC-19, No. 6, pp. 753-768, 1974.

Athanasios Papoulis, *Probability, Random Variables, and Stochastic Processes*, McGraw-Hill, Inc., New York, 1984.

Peyton Z. Peebles, Jr., *Probability, Random Variables, and Random Signal Principles*, McGraw-Hill, Inc., New York, 1987.

Charles W. Therrien, *Decision Estimation and Classification*, John Wiley and Sons, New York, 1989.

INITIAL DISTRIBUTION LIST

1.	Commandant of the Marine Corps Code TE 06 Headquarters, U.S. Marine Corps Washington, D.C. 20380-0001	1
2.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
3.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
4.	Captain Stephen L. Spehn USMC Ground C2 MCRDAC Quantico, Virginia 22134-5080	2
5.	Professor Hal A. Titus, Code EC/Ts Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5002	10
6.	Professor Herschel H. Loomis, Jr., Code EC/Lm Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5002	2
7.	Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5002	1
8.	Commanding Officer Attn: Captain John A. Hucks II USMC Tactical Systems Support Activity MCS Camp Pendleton, California	2
9.	Central Program Facility Attn: LCDR Dan Gahagan SPAWAR 004-51 4555 Overlook Avenue, SW Washington, D.C. 20375-5000	2

10. Naval Research Laboratory — 2
Attn: LT Carol Thompson
Code 9110
Bldg. 259
4555 Overlook Avenue, SW
Washington, D.C. 20375-5000

11. Professor Burl, Code EC/Bl — 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5002

12. Professor Christi, Code EC/Cx — 1
Department of Electrical and Computer Engineering
Naval Postgraduate School
Monterey, California 93943-5002

13. Professor Collins, Code AE/Co — 1
Department of Aeronautical Engineering
Naval Postgraduate School
Monterey, California 93943-5002

14. Professor Hamming, Code CS/Hg — 1
Department of Computer Science
Naval Postgraduate School
Monterey, California 93943-5002